
INTELLIGENCE IN NETWORKS

IFIP - The International Federation for Information Processing

IFIP was founded in 1960 under the auspices of UNESCO, following the First World Computer Congress held in Paris the previous year. An umbrella organization for societies working in information processing, IFIP's aim is two-fold: to support information processing within its member countries and to encourage technology transfer to developing nations. As its mission statement clearly states,

IFIP's mission is to be the leading, truly international, apolitical organization which encourages and assists in the development, exploitation and application of information technology for the benefit of all people.

IFIP is a non-profitmaking organization, run almost solely by 2500 volunteers. It operates through a number of technical committees, which organize events and publications. IFIP's events range from an international congress to local seminars, but the most important are:

- The IFIP World Computer Congress, held every second year;
- open conferences;
- working conferences.

The flagship event is the IFIP World Computer Congress, at which both invited and contributed papers are presented. Contributed papers are rigorously refereed and the rejection rate is high.

As with the Congress, participation in the open conferences is open to all and papers may be invited or submitted. Again, submitted papers are stringently refereed.

The working conferences are structured differently. They are usually run by a working group and attendance is small and by invitation only. Their purpose is to create an atmosphere conducive to innovation and development. Refereeing is less rigorous and papers are subjected to extensive group discussion.

Publications arising from IFIP events vary. The papers presented at the IFIP World Computer Congress and at open conferences are published as conference proceedings, while the results of the working conferences are often published as collections of selected and edited papers.

Any national society whose primary activity is in information may apply to become a full member of IFIP, although full membership is restricted to one society per country. Full members are entitled to vote at the annual General Assembly, National societies preferring a less committed involvement may apply for associate or corresponding membership. Associate members enjoy the same benefits as full members, but without voting rights. Corresponding members are not represented in IFIP bodies. Affiliated membership is open to non-national societies, and individual and honorary membership schemes are also offered.

INTELLIGENCE IN NETWORKS

***IFIP TC6 WG6.7 Fifth International Conference
on Intelligence in Networks (SMARTNET '99)
November 22-26, 1999, Pathumthani, Thailand***

Edited by

Thongchai Yongchareon

*Telephone Organisation of Thailand
Thailand*

Finn Arve Aagesen

*Norwegian University of Science and Technology
Norway*

Vilas Wuwongse

*Asian Institute of Technology
Thailand*



KLUWER ACADEMIC PUBLISHERS

BOSTON / DORDRECHT / LONDON

Distributors for North, Central and South America:

Kluwer Academic Publishers
101 Philip Drive
Assinippi Park
Norwell, Massachusetts 02061 USA
Telephone (781) 871-6600
Fax (781) 871-6528
E-Mail <kluwer@wkap.com>

Distributors for all other countries:

Kluwer Academic Publishers Group
Distribution Centre
Post Office Box 322
3300 AH Dordrecht, THE NETHERLANDS
Telephone 31 78 6392 392
Fax 31 78 6546 474
E-Mail <services@wkap.nl>



Electronic Services <<http://www.wkap.nl>>

Library of Congress Cataloging-in-Publication Data

IFIP TC6 WG6.7 International Conference on Intelligence in Networks (5th : 1999 : Pathumthani, Thailand)

Intelligence in networks : IFIP TC6 WG6.7 Fifth International Conference on Intelligence in Networks (SMARTNET '99) November 22-26, 1999, Pathumthani, Thailand / edited by Thongchai Yongchareon, Finn Arve Aagesen, Vilas Wuwongse. p. cm. — (International Federation for Information Processing)

Includes bibliographical references and index.

ISBN 978-1-4757-1022-9 ISBN 978-0-387-35581-8 (eBook)

DOI 10.1007/978-0-387-35581-8

1. Telecommunications—Data processing—Congresses. I. Yongchareon, Thongchai, 1941– II. Aagesen, Finn Arve. III. Wuwongse, Vilas. IV. Title. V. International Federation for Information Processing (Series) ; 30.

TK5101.A1 I334 1999
621.382—dc21

99-047404

Copyright © 2000 by International Federation for Information Processing.

Softcover reprint of the hardcover 1st edition 2000

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, mechanical, photo-copying, recording, or otherwise, without the prior written permission of the publisher, Kluwer Academic Publishers, 101 Philip Drive, Assinippi Park, Norwell, Massachusetts 02061.

Printed on acid-free paper.

CONTENTS

Preface.....	ix

Program and Organizing Committees.....	xi

Keynote Session

Active Networks Nodes for Adaptive Multimedia Communication <i>Otto Spaniol and Jens Meggers</i>	1

Internet Based Service Development <i>Olli Martikainen and Jouni Karvo</i>	19

Smart vs. Intelligent vs. Active IP Networks <i>Guy Pujolle</i>	27

Invited Paper Session

ATM Security: A Survey of Research, Products, and Standards Extended abstract of the mini-tutorial <i>Herbert Leitold</i>	35

Mobile Agents and Active Networks <i>Fritz Hohl</i>	45

Dependability Issues in Smart Networks <i>Bjarne E. Helvik</i>	53
Service Feature Interaction <i>Elie Najm</i>	77
Technical Session: Teleservices - 1	
Developing a Distributed Meeting Service to Support Mobile Meeting Participants <i>Seamus Hickey</i>	81
Voice Services for PSTN and IP Networks <i>Qi Guan</i>	95
INexpensive Open Distributed Service Platform <i>Tor Martin Didriksen, Lars Sivert Sørungård and Tor Øystein Molnes</i>	107
Technical Session: Application of Mobile Agents - 1	
A Multiagent System Architecture for Quality of Service Management in Distributed Multimedia Systems <i>Mamadou Tadiou Kone and Tatsuo Nakajima</i>	121
Mobile Agents for Mobile Users <i>Do van Thanh, Sverre Steensen and Jan A. Audestad</i>	135
A Mobile Agent-based Approach for the UMTS/VHE Concept <i>Peyman Farjami, Carmelita Görg and Frank Bell</i>	149
Technical Session: High Capacity Networks and ATM	
A Novel Efficient and Robust Explicit Rate Congestion Avoidance Algorithm for ABR Service in ATM Networks <i>Tanun Jaruvitayakovit, Naris Rangsinoppamas, Boonchoung Tansuthepverawongse and Prasit Prapinmongkolkarn</i>	163
A Distributed Routing Algorithm in Clos Network of Variable Bit Rate TDM Switch <i>Hiroaki Morino, Hitoshi Aida and Tadao Saito</i>	177

ATM Network Security:

Requirements, Approaches, Standards, and the SCAN Solution

Herbert Leitold and Reinhard Posch 191**Technical Session: Teleservices - 2****Unified Messaging:**

A Framework Approach for building messaging services

Timo HäMäläinen and Esa Saukkonen 205Connection and Location Management based on a TINA-compliant
Architecture for UMTS*Hoang Nguyen-Minh and Harmen R. van As* 219**Technical Session: Middleware and Specification Languages**

Relationship between SDL Processes and Message Sequence Charts

Jie Lian and Vilas Wuwongse 237

A Trading Model of Stream Binding Selection

F. Eliassen and H.O. Rafaelsen 251

Using Roles with Types and Objects for Service Development

Rolv Bræk 265**Technical Session: Teleservices - 3**A Reference Model for Conceptualising the Convergence of
Telecommunications and Datacommunications Service Platforms*Pasi Kemppainen* 279**Technical Session: Teleservices - 4****The Wandering Logic of Intelligence:**

Or Yet Another View on Nomadic Communications

Plamen L. Simeonov 293

Plug-and-Play Network Service Configuration Using CORBA

Syed Kamran Raza, Bernard Pagurek and Tony White 307

Towards a Plug and Play Architecture for Telecommunications

*Finn Arve Aagesen, Bjarne E. Helvik, Vilas Wuwongse,
Hein Meling, Rolv Bræk and Ulrik Johansen* 321

Technical Session: QoS and Performance Management

Distributed QoS Management for a Multiparty Video Conferencing System <i>Ines Beier and Hartmut Koenig</i>	335
Performance Evaluation of an IP Voice Terminal <i>Harri Marjamäki and Raimo Kantola</i>	349
On Load Balancing in Distributed Telecommunication Systems <i>Per-Oddvar Osland and Peder J. Emstad</i>	363

Technical Session: Teleservices - 5

Enabling Factors for IN Service Provider Access <i>A.Jøsang, M.G.L.Bijnagte and G.Kleinhuis</i>	375
An Experimental Environment for Distributed Intelligent Network Services <i>Bilhanan Silverajan and Jarmo Harju</i>	389
From SIBS to Distributed Objects: A Transformation Approach for Service Creation <i>Elie Najm, Frank Olsen and Sylvie Vignes</i>	403

Technical Session: Application of Mobile Agents - 2

Multi-Agent Support for Modelling Co-operative Work <i>Mihhail Matskin</i>	419
The Use of Mobile Agent based System as a Solution for Efficient Management of Telecommunication Networks and Services in a Multi-domain Environment <i>Nazim Agoulmine, José Neuman de Souza, Jorge Luis de Castro e Silva and Arnaldo Nunes da Silva</i>	433
Mobile Agents in Implementation of a Virtual Home Environment <i>Kirsi Valtari</i>	445
Author Index	459
Index	461

PREFACE

SMARTNET'99 is the fifth in a series of conferences on Intelligence in Networks under the auspices of IFIP TC6's Working Group 6.2. "Grade of Network Intelligence" can be defined as the *efficient flexibility in the introduction of new teleservices*. Intelligent Networks, Telecommunication Information Networking Architecture (TINA), Mobile Agents and Active Networks are different solutions to improve the grade of network intelligence. "Smart Networks" is here used as an umbrella concept for various solutions to improve the network intelligence.

The main objective of SMARTNET'99 is to provide an international forum for academics, researchers, practitioners and services providers. The discussion areas include the research and development within various closely related fields such as: telecommunication services and service architectures, service creation, platforms/middleware, specification languages, mobile agents, active networks, plug-and-play issues and Quality of Service. These topics are covered by 15 sessions. There are 5 sessions with keynote and invited papers, and 10 technical sessions. The papers in the 10 technical sessions are arranged in five sessions on "Teleservices", two sessions on "Application of Mobile Agents", and one session on each of the following: "High Capacity Networks and ATM", "Middleware and Specification Languages" and "QoS and Performance Management".

Before the conference, a two-day tutorial has been arranged with the following topics: "Teleservice Architectures and Platforms", "Internet", "Specification and Verification of Protocols", "SDL2000", "Security and

Electronic Commerce”, “Mobile Agents and Active Networks” and “From IP over ATM to MPLS, A new Internet Architecture”.

SMARTNET'99 has four keynote and five invited papers. The keynote speakers and their topics are: Professor Otto Spaniol from the University of Karlsruhe in Germany: “Active Network Nodes for Adaptive Multimedia Communication”, Professor Olli Martikainen from the Helsinki University of Technology in Finland: “Internet Based Service Development”, Professor Guy Pujolle from the University of Versailles in France: “Smart vs. Intelligent vs. Active IP Networks” and Dr. Thaweesak Koanantakool from the National Electronics and Computer Technology Center in Thailand: “Software Park for Thailand”. The Invited speakers and their topics are: Mr. Herbert Leitold from Graz University of Technology, Austria: “ATM Security: A Survey of Research, Products and Standards”, Mr. Fritz Hohl from University of Stuttgart, Germany: “Mobile Agents and Active Networks”, Professor André Danthine from the University of Liège in Belgium: “How to Provide QoS in the Next Generation Internet?”, Professor Bjarne Helvik from the Norwegian University of Science and Technology in Trondheim, Norway: “Dependability Issues in Smart Networks” and Dr. Elie Najm from Ecole Nationale Supérieure des Télécommunications (ENST) in Paris, France: “Service Feature Interaction”.

Many people have contributed to the organisation of this conference. We thank the members of the Program Committee for their continuous advice and help in reviewing and selecting papers. We would also like to thank IFIP TC6 and the sponsors for their support of this conference. Finally we would like to express our gratitude to the Organising Committee chaired by Dr. Kosol Petchsuwan for their excellent work and also Ms. Chutiporn Anutariya for her excellent technical advice and assistance during the whole process from "Call for Papers" to the technical preparation of this Proceedings.

Thongchai Yongchareon, Finn Arve Aagesen and Vilas Wuwongse

PROGRAM AND ORGANIZING COMMITTEES

Program Committee Chairpersons:

Thongchai Yongchareon, Thailand (Chairman)
Finn Arve Aagesen, Norway (Vice-Chairman)
Vilas Wuwongse, Thailand (Vice-Chairman)

Program Committee Members:

Pakpong Akaniwan, Thailand
Piergiorgio Bosco, Italy
Augusto Casaca, Portugal
A. Lyman Chapin, USA
André Danthine, Belgium
Domenique Gaiti, France
Trevor Housley, Australia
Terje Jensen, Norway
Somnuk Keretho, Thailand
Dipak Khakhar, Sweden
Thaweesak Koanantakool, Thailand
Olli Martikainen, Finland
Chutichai Napasab, Thailand
Reinhard Posch, Austria
Prasit Prapinmongkolkarn, Thailand
Guy Pujolle, France
Harry Rudin, Switzerland

Sutus Rukspollmuang, Thailand
Tadao Saito, Japan
A.B. Sharma, Thailand
Booncharoen Sirinaovakul, Thailand
Otto Spaniol, Germany
Anand Sudarshan, India
Booskorn Tanasomboonkit, Thailand
Prakit Tangtisanon, Thailand
Samir Tohme, France
Kitin Udomkiat, Thailand

Organising Committee Chairpersons:

Kosol Petchsuwan, Telecommunications Association of Thailand
(Chairman)
Kristian Bo, TelecomAsia (Vice-Chairman)

Organising Committee Members:

Tapio Erke, Asian Institute of Technology
K.K. Gunawardena, Asia-Pacific Telecommunications
Thammanoon Julmanichoti, Communications Authority of Thailand
Pleumjai Sinarkorn, Thai Telephone & Telecommunication
Sittichai Songpiriyakit, Telephone Organization of Thailand
Sukit Tirawatanawit, Shinnawatra Group

Organisers:

Asian Institute of Technology (AIT)
International Federation for Information Processing (IFIP)
Telecommunications Association of Thailand (TCT)

Sponsors:

Alcatel (Thailand) Co., Ltd.
Ericsson Communications (Thailand) Ltd.
Lucent Technologies (Thailand) Inc.
Nokia Telecommunications (Thailand) Ltd.
Siemens Systems Ltd.
ECI Telecom Ltd.
Mitsui & Co., Ltd.
NEC Corporation

ACTIVE NETWORKS NODES FOR ADAPTIVE MULTIMEDIA COMMUNICATION

Otto Spaniol, Jens Meggers

*RWTH Aachen, Department of Computer Science 4
Ahornstr. 55, 52056 Aachen, Germany*

Abstract: This paper proposes the use of active networking technology for congestion control and fast error correction with application level error recovery code deployed in active nodes. Its focus lies on real-time video transmission of sender based adaptive applications which are supported by active network nodes. Experiments with a prototype model show a considerable improvement with respect to video quality, error correction time, and bandwidth utilisation

1. INTRODUCTION

Active networks introduce a new powerful communication paradigm that provides substantial benefits for multimedia communication: nodes within the network become able to execute application or user specific code that may introduce new protocol elements or process incoming data streams. This enables network nodes to adapt traffic by application specific methods which are installed in network nodes, or even are part of the transported packet streams. This promising approach has a great deal of unexplored potential: applications are freed from complex processing and new high-level protocols are dynamically integrated without changing network software. Furthermore, new user specific adaptive algorithms become possible which increase the flexibility of multimedia communication systems.

A highly flexible multimedia communication system is of great importance when considering the requirements of upcoming applications, services, and heterogeneous networks. Multimedia communication requires

high bandwidth, low packet delay, low jitter, and also small packet loss rates. In particular, mobile multimedia communication demands flexible resource management and highly adaptive data streams that are able to adapt to varying packet error rates and sudden network congestion caused, e.g., by terminal mobility.

In addition to applications like network management or dynamic protocol deployment, two still existing problems in multimedia communication can be addressed by means of active network nodes: congestion control at network bottlenecks and application level error recovery. Dedicated application-specific procedures within a set of network nodes enable stream-specific traffic shaping. Instead of adapting the transmission throughput at the transmitting application, congestion can better be directly controlled at the network bottleneck, leading to fast bandwidth adaptation and minimal packet loss. Especially in heterogeneous networks, bottlenecks arise at gateway routers that connect different network technologies, e.g. wireless and fixed networks. Here, congestion control by active network nodes can contribute to improved interactive real-time multimedia communication.

This paper describes how active network nodes can support both congestion and error control for real-time video streams. We show how sender based adaptive applications can benefit from active nodes and describe a traffic reduction and error control scheme that substantially increases the overall end-to-end video communication quality.

2. CONGESTION CONTROL

Interactive real-time multimedia streams have strong requirements concerning packet loss, end-to-end delay, and jitter. As the Internet does not support these type of applications, different solutions have been proposed that promise the reduction of network congestion and consequently reduce packet loss and end-to-end delay.

Sender Adaptation

Sender based adaptation requires adaptive applications [DHT95] that adapt their output rate to the present network condition. The Direct Adjustment Algorithm [Sis97] for example decreases the senders' data rate when detecting network congestion. In case of a non-congested state, the sender increases its transmission speed periodically until a congestion is detected. The scheme is designed analogous to the TCP flow control scheme. Some schemes claim to be TCP-friendly [RHE99], which means that the available bandwidth is shared between TCP and the adaptive multimedia data streams without letting TCP streams dominate the others, or vice versa. The main disadvantage of the adaptive sender approach is the fact that the sender is only notified after the reception quality has been

deteriorated. Thus, redundant coding becomes necessary that allows to compensate for a certain amount of packet loss. Another drawback is that the sender would have to adapt its transmission rate to the available bandwidth of the weakest receiver when multicast transmission is used. In scenarios with a high number of participants this would lower the overall delivery quality even for users connected by a reliable high speed link.

Receiver Adaptation

The receiver based adaptation approach [MJV96] has been designed to solve the multicast disadvantage of the sender based adaptation scheme. Here, the receiver simulcasts separate streams of different quality layers of the transmitted stream. The receiver detects packet losses and subscribes or un-subscribes to certain layers in order to meet a specific throughput. The receiver based approach has shown to perform well in multicast scenarios but is limited by the number and granularity of the separated sub-streams. The multimedia content needs to be very flexible in order to allow the splitting into many different sub-streams. Although latest video compression techniques like H.263+ [Côt98] enable multi-layered coding, the encoding process becomes more complex and produces more overhead.

Resource Reservation

When using resource reservation, applications signal their delay and throughput requirements to the appropriate network nodes and receive an admission control acknowledgement [FRV92] [CSZ92]. In case of a negative acknowledgement they have to reduce the desired resources and start a new request. To guarantee a certain delay or packet loss rate, applications would have to reserve their peak data rate, leading to poor network utilisation. Sophisticated measurement based algorithms [JDS95] try to increase the utilisation by allowing mean rate allocation. However, these measurement based schemes only work at a high amount of statistical multiplexing that is not given in some networks, e.g. mobile networks. Another disadvantage is that even measurement based approaches do not reach a utilisation above 80% and cannot avoid packet losses or delay violations. A further practical disadvantage of resource reservation is its impracticability in wireless networks where end-systems move and the error characteristic of the wireless transmission medium may change over time. The network's resource reservation would have to track the movement of the mobile terminal, leading to unpredictable resource availability. Although some schemes exist that allow admission control for mobile users [TAL97], the call blocking probability increases drastically with rising network utilisation.

Priority Systems

Priority systems such as type of service routing or the differentiated services approach [Bla98] allow for service differentiation and enable the deployment of packet schedulers that give real-time packets priority over

other service classes [FIJ95]. Despite the usefulness of such systems, the congestion problem is still present within a single service class.

Active Networks

A solution that makes use of active network nodes is to adapt the upcoming traffic directly at the congested node. This can be done by installing dedicated procedures that are able to reduce the required bandwidth in case of a congestion within a network node. The required adaptation procedures can be transported by the multimedia stream itself in order to allow stream specific adaptation. A simple active network could have fixed, pre-installed adaptation procedures that are selected according to the payload type or the congestion state. The active network solution is very interesting because it works well in multicast systems, allows for high bandwidth utilisation, and can avoid packet loss and delay violation. However, such a system comes at a cost: a complex network infrastructure is needed; additional security problems arise; new applications become necessary and the active nodes have to consider an additional spare resource - CPU time. Nevertheless, there are many application where the advantage of such a solution outweigh the disadvantages. For example, a wireless cellular network consists of many small cells, each serving only few users. Here, the additional required CPU resources would not be too high and would allow to avoid packet loss and delay violations.

In order to allow for fair link sharing and limited CPU usage at the active node, bandwidth adaptation by means of active network nodes can be combined with the above mentioned congestion schemes. For example, a resource reservation system can make use of an active network in order to implement dynamic traffic filters [WiZ98], or a differentiated service system can deploy active network elements for traffic shaping and metering.

Network support for applications that make use of end-to-end flow control and sender based adaptation schemes is another promising approach. Here, active nodes could be placed at selected network bottlenecks to avoid a decreasing application level communication quality due to packet loss. In the following, this approach is examined in more detail.

3. THE ADAPTIVE ACTIVE NETWORK APPROACH

The approach presented in this paper is to install active nodes at network bottlenecks to increase the application level quality of service of adaptive applications. An adaptive application that transmits video streams controls its output rate by adjusting the video picture quality according to the filling grade of the transmitters' output queue (local feedback) [DHT95]. Receivers

periodically send packet loss reports to the sender application that calculates a maximum output rate and configures its transmitter appropriately (receiver feedback).

Figure 3.1 shows the combination of an active network node with an adaptive application. By installing an active network node in the transport path from the transmitter to the receiver, short term network congestion control and error recovery can be done without affecting the participating applications. The active node can predict congestion by monitoring the number of packets that reside in its input or output buffers (congestion prediction). When necessary, the active node starts application specific procedures that reduce the amount of data per stream.

We propose to install active network nodes at network routers which form bottlenecks. These routers connect different network types, e.g. a core network of a cellular network to the Internet.

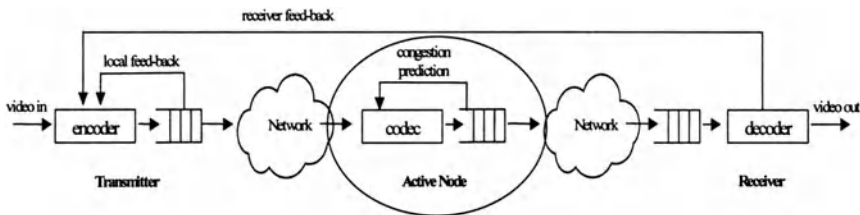


Figure 3.1. Deploying Adaptive Applications with Active Nodes

An active network has many advantages that help to install and initiate application specific adaptation procedures within the network. It allows to install transcoder units that may (dynamically) reduce the amount of bandwidth consumed by any given data stream. As these procedures depend on the respective application, active networks provide a flexible framework enabling the installation and initiation of user specific adaptation procedures within the network. In addition, active network nodes may initiate error recovery processes that are specific to the transported streams. For example, a multicast delivery of a newspaper would require packet re-transmission in case of a packet loss, while a real-time transmission of a video stream would require to update the video frame content of the following frame.

In the following section a sample implementation of an active node that operates on H.261 video streams is shown. Although the described prototype implementation processes H.261 video streams, the results should be viewed independent of this coding technology. Similar results can be achieved with other encoding schemes and algorithms. It is more important to understand that the active node's process depends on the application and the user preferences. Consequently, it is of great importance that the active node can

be (dynamically) configured to support other adaptation processes depending on user and application requirements.

4. CONGESTION CONTROL

In order to investigate congestion control by active network nodes, we prototyped a sender based adaptive application and an active node. The sender transmits intra coded H.261 [H261] video streams with a conditional replenishment algorithm via RTP [Sch96] as implemented in the vic and IVS system [MCC95][TUR96]. End-to-end bandwidth adaptation is done by the Direct Adjustment Algorithm (DAA) [Sis97]. The DAA algorithm increases the output rate by an Additive Increase Factor (*AIF*) when the reported loss remains below a certain loss threshold. When the loss rate is reported above a pre-defined threshold, the output rate is reduced by the loss-rate reported from the receiver.

The transmitter configures its throughput to the value estimated by the DAA scheme by changing the quantisation parameter of the H.261 encoder [BoT94]. Because the resulting output bandwidth is of variable bitrate, it is difficult for a sender to exactly meet an envisaged output rate. Thus, only the mean output rate of the video transmitter is controlled, sometimes leading to an output rate above the calculated value. According to our measurements this may cause unfairness. We modified the DAA scheme to calculate the new output bitrate from the actually measured rate if the measured output rate is higher than the envisaged output rate. This modification ensures that a source transmitting at a rate higher than the one given by the DAA scheme is reduced by a fraction of the actual transmitted data. The modified DAA scheme is as follows:

```

▪ If (reported_loss_ratio > loss_threshold)
▪   If (measured_output_rate > output_rate)
▪     output_rate = max(min_rate , measured_output_rate ·
      (1- reported_loss_ratio + loss_threshold))
▪   Else
▪     output_rate = max(min_rate, output_rate · (1-
      reported_loss_ratio + loss_threshold))
▪ Else
▪   output_rate = min(max_rate, output_rate + AIF)

```

We implemented the active node as an RTP translator [Sch96] with a single output queue. The packets are serviced by a FCFS scheduling discipline. Figure 4.1 shows the experimental set-up. Four video transmitters send video streams through the active node to a corresponding destination.

The transmitters are interconnected by a 100 Mbit/s Fast Ethernet and the output of the active node is configured to transmit with a maximum output rate of 1,5 Mbit/s. Thus, the four sources will have to share 1.5 Mbit/s. The minimum transmission rate of each transmitter is set to 228 kbit/s (*min_rate*) and the maximum to 1,5 Mbit/s (*max_rate*). The AIF is configured to 50 kbit/s. The loss threshold is set to 5 %.

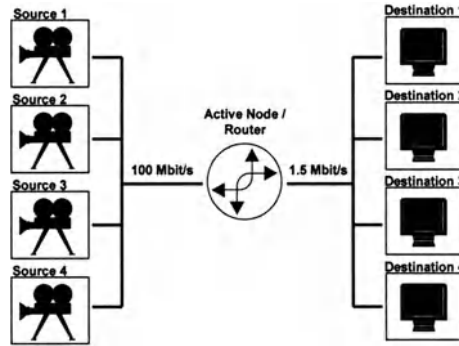


Figure 4.1. Experiment Configuration

Figure 4.2 shows how the bandwidth is shared by the 4 video sources. Every 3 minutes, another source is started. It can be seen that the bandwidth is equally shared and that newly incoming streams force the other sources to reduce their bandwidth usage.

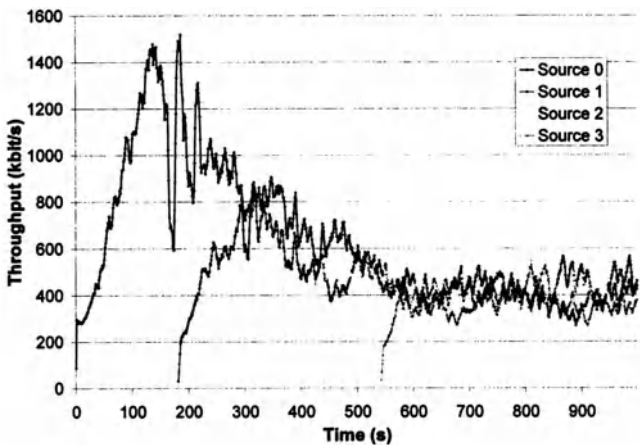


Figure 4.2. Bandwidth share controlled by the DAA scheme

We have implemented a simple bandwidth reduction scheme at the active node that does H.261 packet by packet transcoding. Although we transcode the incoming stream from H.261 to H.261, we have chosen the transcoding approach in order to be more flexible in adapting bandwidth, e.g. by changing the frame-rate, video colours, or by a picture size reduction.

We have chosen the DAA for end-to-end bandwidth adaptation in order to relieve the active node from instantly reducing the throughput. When all participating sources transmit with a constant bit rate, the estimated throughput of n sources can exceed the maximum available bandwidth by $n \cdot AIF$ plus the loss threshold fraction of the maximum available bandwidth. Thus, the active node would only have to reduce the amount of bandwidth used per stream by the AIF and a threshold fraction of its actual rate. However, this calculation does not hold for sources with variable bitrate where bandwidth peaks may overlap. Thus, we have configured the reduction process of the active node by multiplying the prevailing quantisation of each connection by three when the output queue length of the active node exceeds 1/3 of its maximum size (the maximum size was set to 100 packets). This reduced the data rate of the processed connection by more than the AIF , but on the other hand it limits the execution time for the transcoding process.

An important implementation detail of the active node is the additional delay due to packet processing. In our implementation, re-quantisation is done by transcoding on a per packet basis. This adds an additional worst case end-to-end delay of one packet decoding and encoding time, plus one packet inter-arrival time in case of packet re-segmentation. The packet decoding and encoding duration depends on the amount of macro blocks that are stored in a packet, which in turn depends on the encoder's quantisation parameter. As shown in table 4.1, the decoding duration varies between 0.13 and 0.94 ms and the encoding duration between 0.15 and 5.2 ms. The encoding and decoding duration becomes significantly higher when the DCT/IDCT is processed. However, when processing DCT/IDCT, the transcoding process is not limited to a simple re-quantisation task. Instead, it would be possible to implement colour or size reduction of the video pictures.

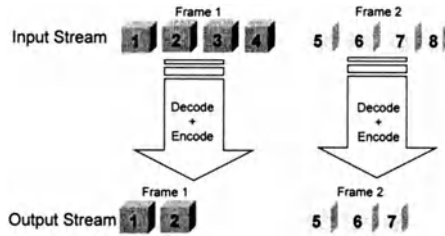


Figure 4.3. Packet by Packet Transcoding/Re-quantisation

Our implementation re-segments packets in order to reach packets sizes equal to the maximum transfer unit (MTU), to achieve the maximum performance on the destination network, and to limit packet overhead. Packets that are reduced by the re-quantisation process will be detected as packet loss at the receiver, because the active node resynchronises the RTP sequence number with the next arriving video frame. This allows end-to-end schemes like the DAA to work in conjunction with the active node. Figure 4.3 shows the packet by packet transcoding scheme.

In order to evaluate the improvement of our system, we were interested in application level quality metrics. We selected the Peak Signal to Noise Ratio (PSNR) value and the video frame inter arrival time at the receiver for comparing the active node operation with a simple FIFO packet scheduler. Both versions deploy the DAA scheme on an end-to-end basis.

Table 4.1. Decoding and encoding duration for a single packet

Quant	Blocks per Packet	Decoding IDCT	Decoding no IDCT	Encoding DCT	Encoding no DCT
30	66.7	0.94 ms	0.7 ms	5.2 ms	0.95 ms
15	44.1	0.82 ms	0.4 ms	3.7 ms	0.65 ms
1	4.5	0.24 ms	0.13 ms	0.47 ms	0.15 ms

To measure the frame inter arrival deviation at the receiver, we have measured the time between two consecutive video frames at the sender and the corresponding inter arrival time at the receiver. The difference of this value for frame i gives us the video frame jitter Δj_i . Figure 4.4 shows a histogram of Δj_i measured on the basis of 10000 delivered frames. The experiments have shown that the video frame inter arrival time of the active node operation shows smaller jitter values than the simple system with a FIFO packet queue. The mean Δj_i for the DAA with FIFO queuing is 32 ms, and the average jitter for the system with active network support is 24 ms. Thus, the receiving application needs to maintain a smaller jitter buffer that directly reflects the end-to-end communication delay. The lower video frame inter arrival times are caused by smaller queuing delays in the active

node queue because the overall packet delay is reduced by the lower average filling grade of the output queue of the active node. Another important factor is that the active node operation avoids packet loss. Instead, packets are reduced by transforming the video data to a lower quality level. Because of the segmentation of video frames on RTP packets, as described in RFC 2032, the receiving application will have to wait for the last packet belonging to the actual frame before displaying the video frame. When the last packet of a video frame is lost, the receiver cannot display the frame until a packet of the next frame arrives¹.

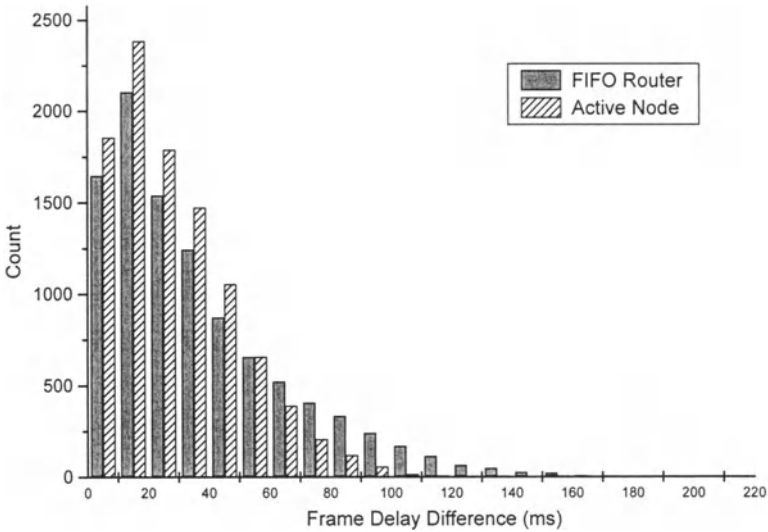


Figure 4.4. FIFO vs. Active Node Video Frame Interarrival Times

A second quality metric measured on application level is the PSNR of the video at the receiving application. Two processes affect the actual video quality. One is the sender application that configures the video encoder according to the aspired output rate. The other is the active node that lowers the video quality when the transcoding process is started.

Our prototype implementation measures the PSNR both at the receiver and the sender application simply by comparing the transmitted and received stream with the original video signal that has been stored to a file. Thus, the ΔPSNR_i for the transmitted video frame i , which is a measure for the degradation of video quality during transmission, can be calculated as:

¹ This kind of implementation assumes that packet reordering does not take place.

$$\Delta PSNR_i = 20 \log_{10} \frac{255 \cdot N^2}{\sqrt{\sum (T(i, j) - F(i, j))^2}} - 20 \log_{10} \frac{255 \cdot N^2}{\sqrt{\sum (R(i, j) - F(i, j))^2}}$$

where $F(i, j)$ is the input signal at the sender's encoder, $T(i, j)$ the output signal from the encoder, and $R(i, j)$ the signal received at the receiver application. $R(i, j)$ differs from $T(i, j)$ when the active node performs bandwidth adaptation or when packets are lost.

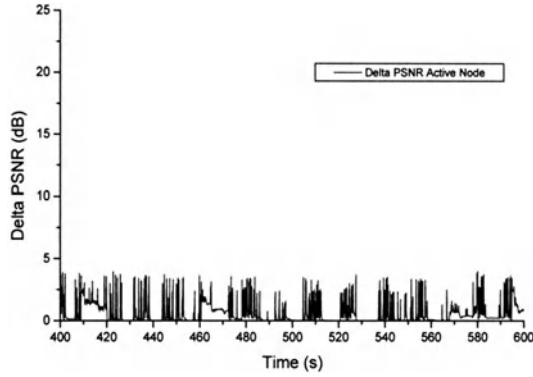


Figure 4.5 a. $\Delta PSNR$ with the Active Node

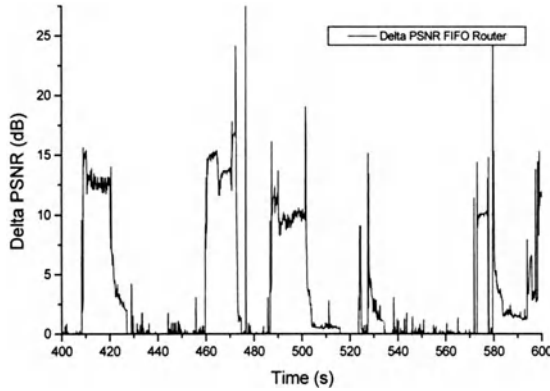


Figure 4.5 b. $\Delta PSNR$ with the FIFO Router

Due to the fact that the active node reduces the video quality in case of congestion, the receiving video quality is lower than the quality at the

sender. However, without the active node operation, packets are dropped when the queue reaches its maximum capacity. Thus, the quality without the active node is much lower than with the active node operation. Figure 4.5b shows the Δ PSNR measured at the sending and receiving application when the DAA and a FIFO queue is used. Here, the mean Δ PSNR is 1.98 dB with a standard deviation of 3.88 dB.

In contrast, the active node allows for a lower Δ PSNR as depicted in Figure 4.5a. Here, the average Δ PSNR is about 0.48 dB with a standard deviation of 0.84 dB. Figure 4.5a shows that the active node achieves a considerable improvement in quality.

5. ERROR CONTROL

The reason for the high Δ PSNR values of the DAA scheme without active node adaptation is as follows: when a packet from a video stream is lost, the corresponding display area will be damaged until a subsequent video frame encodes the macro blocks in intra mode. When the encoder deploys motion compensation with motion vectors, the display errors will extend over time and space. With MPEG, the error will not be corrected until the next I-frame is transmitted. The ITU standards H.261 and H.263 do not dispose the repetition of a video frame encoded in full intra frame mode. In order to allow for minimal end-to-end delay, the frame sequence can be encoded in P-frames that may skip a macro block or encode it in P- or I-mode. This feature reduces the overall throughput and avoids peaks that may cause congestion or delayed transmission. However, long I-frame repeating intervals make it necessary to deploy a feedback scheme that triggers the sender's encoder to correct display errors by refreshing spatial video regions in intra mode² [MeS99].

In [BoT98] and [MeS99], such feedback schemes have been examined. The scheme proposed in [MeS99] reports the actual state of its display to the sender by mapping the timestamps of the received packets to the corresponding display region. It has been shown that this scheme introduces only very few additional bandwidth usage [MeS99].

Instead of an end-to-end error feedback control as analysed in [MeS99], we have implemented hop by hop error recovery between active nodes. When the active node receives an error report from either the receiving application or from another active node located downstream, it inserts the

² The area of blocks to be encoded in intra mode depends on the usage and the distance of motion vectors.

contents of the lost video regions into the next video frame. Figure 5.1 shows the configuration of our with 2 active nodes.

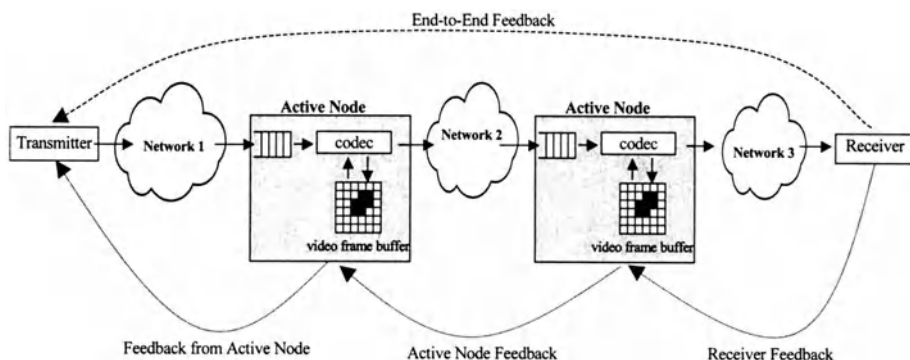


Figure 5.1. Error Control with End-To-End Feedback (Top) and Active Node Feedback (Bottom)

In order to evaluate the improvement of such a system, we installed the configuration as depicted in figure 5.1. We connected the different systems by a LAN and introduced additional queuing delay in the output queues of the transmitter and the active nodes in order to measure results comparable to a wide area network. In addition, we implemented an error simulation within the input queues of the receiver and the active nodes. The error simulation model uses a Gilbert model to simulate burst errors. This allows us to conduct experiments with different error rates. To test the behaviour of the active node error recovery within an heterogeneous network, we configured the propagation delay and the packet error rates as summarised in table 5.1. Whereas the first two networks each introduce a queuing delay of 100 ms and a packet error rate of 10 %, the last hop represents an access network a smaller propagation and queuing delay. This configuration could represent a wireless access network connected to a wired wide area network (e.g. the Internet).

Table 5.1. Experimental Queuing Delay and Packet Error Rates

	▪ Network 1	▪ Network 2	▪ Network 3
▪ Packet Error Rate	▪ 10 %	▪ 10 %	▪ 20 %
▪ Delay (ms)	▪ 100 ms	▪ 100 ms	▪ 20 ms

We compared the end-to-end feedback with the feedback between active nodes. After sending 4000 video frames, we observed an average Δ PSNR of 2.35 dB per video frame when deploying end-to-end feedback error control.

The hop by hop feedback scheme between active nodes and applications could improve the video quality and resulted in a Δ PSNR of 1.81 dB. Another measure for video quality is the number of macro blocks that appear on the receivers display without error. Whilst the average number of damaged macro blocks is 48,22 for the end-to-end error correction, the active nodes reduce this number by 27 percent to 35 blocks (see Table 5.2).

Table 5.2. Video Quality of the different Error recovery Configurations

	Average Δ PSNR	Damaged Macro Blocks per Frame
No Error Correction	11,72	143,25
End To End Error Correction	2,35	48,22
Active Node Error Correction	1,81	35,01

Figure 5.2 shows the load generated on the three connecting networks by the single source. Whilst the load of the end-to-end feedback version is reduced at each node by the packet error rate, the load when deploying active nodes is reduced less due to the error correction done by the intermediate nodes. Compared to the end-to-end error correction, the active network error correction reduces the network load in network 1 and network 2 and adds a small amount of additional bandwidth within network 3. However the total network load of the active network system is lower than that of the end-to-end feedback scheme. Despite of this lower overall bandwidth consumption the quality of the video is improved.

6. DISCUSSION

Despite the benefits of active network usage for congestion and error control as described in the former sections, some questions remain that have to be solved before deploying this type of network system.

One remaining problem is the CPU resources required at the active node. Although the adaptation process will only adapt the data steams during periods of congestion, the required CPU usage during these intervals increases linearly with the number of transmitted streams. For a transcoding approach as described in the former chapter this will considerably limit the number of transported streams. However, modern coding techniques such as Wavelet coding or hierarchical coding in H.263+ and MPEG-4 promise to simplify the transformation process by simply dropping packets of lower

coding layers. Nevertheless, the active code running in the active node is a further resource that has to be admitted and charged. One approach is proposed in the SwitchWare active network model [Ale98a]. Here, the CPU cycles per user specific code can be measured and restricted.

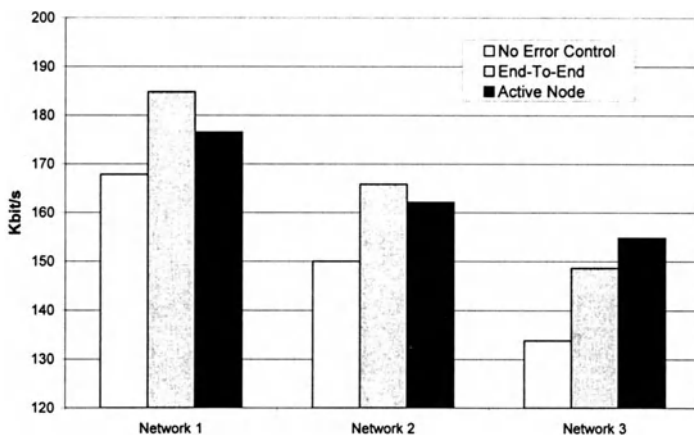


Figure 5.2. Network Load caused by the end-to-end Corrections Compared to Active Network Correction

A further argument against the deployment of active network nodes is the so called *end-to-end argument*, that says not to implement high level functionality within the network if it can be implemented better and more cost-effectively in end systems, especially if some applications may not benefit from such functions at all [SRC84][Bha97]. However, in [Bha97] it is shown that the end-to-end argument does not rule out high level functionality within networks. Nodes within the network may have information that is not available in end systems, such as congestion states or traffic load pattern. Thus, the combination of application level information, e.g. payload dependencies or data priority, may provide for services that can be best supported or enhanced only if application and network information are combined. Due to active networks inherent property of being application driven, and since services can be tailored to the user's requirements, the active network can be implemented cost effectively, because it is only used by applications that have configured them to their needs.

Another important consideration deals with security aspects of active networks. In addition to many security problems that arise when installing and maintaining an active network [Ale98b], it is a basic requirement from the user's point of view that the active network only allows those code segments to process the applications' data stream that have originated from the application itself. If the code has been not generated by the application, it

has to be ensured that this code is a well known or certified one. Consequently, the active network system has to ensure that the transported or invoked code is not modified prior to or during execution. A further security problem is the active network support for encrypted data streams.

7. CONCLUSION

We have introduced an active network node that avoids network congestion by decreasing the video quality of transmitted video streams in periods of congestion. The active node works on the application level and thus allows to apply application specific adaptation procedures to the transported data. Our implementation makes use of a specific video transcoder to demonstrate the benefits of application level bandwidth adaptation by means of active network nodes. A sender based adaptation scheme was combined with an active network node that allows to minimise the adaptation process at the router and to perform end-to-end flow control.

We have shown that the network adaptation approach can significantly improve the end-to-end video quality, and can contribute to error correction, required when packets are dropped in network nodes that connect different active nodes.

This work has shown that application specific procedures installed within the network can significantly increase the overall service quality. Installed in a secure active network environment, application specific congestion and error control procedures may enable interactive real-time communication even in unreliable, congested networks.

8. REFERENCES

- [Ale98a] D. S. Alexander et al, "*The SwitchWare Active Network Architecture*", IEEE Network Special Issue on Active and Controllable Networks, Vol. 12 No. 3, pp. 29 – 36, IEEE Press 1999.
- [Ale99] D. S. Alexander et al, "*Security in Active Networks* ", Secure Internet Programming: Issues in Distributed and Mobile Object Systems, Springer-Verlag Lecture Notes in Computer Science State-of-the-Art series.
- [Bha97] S. Bhattacharjee et al, "*Active Networking and the End-to-End Argument*", ICNP '97, Atlanta, GA, October 1997.
- [Bla98] S. Blake et al, "*An Architecture for Differentiated Services*", RFC 2475, December 1998.
- [BoT94] J. Bolot , T. Turetli, "*A rate control for packet video in the Internet*", Proc. IEEE INFOCOM '94, pp. 1216-1223, Toronto June 1994.

- [BoT98] J-C. Bolot, T. Turletti, "*Experience with Control Mechanisms for Packet Video in the Internet*", ACM SIG COMM Computer Communication Review, Vol. 28, No. 1, pp. 4-15, January 1998.
- [Côt98] G. Côté et al, "*H.263+: Video Coding at Low Bit Rates*", IEEE Transactions on Circuit and Systems for Video Technology, November 1998
- [CSZ92] D. Clark, S. Schenker, and L. Zhang, "*Supporting Real-Time Applications in An Integrate services packet network: Architecture and Mechanisms*", Proc. SIGCOMM'92, 1992.
- [DHT95] C. Diot, C. Huitema, T. Turletti, "*Multimedia Applications should be Adaptive*", Proc. HPCS'95, Mystic (CN), Aug. 23-25, 1995
- [FRV92] D. Ferrari, J. Ramaekers, G. Ventra, "*Client-Network Interaction in Quality of Service Communication Environments*", 4th IFIP Conference on High Performance Networking, University Liège, Belgium, 1992.
- [FIJ95] S. Floyd, V. Jacobson, "*Link-sharing and Resource Management Models for Packet Networks*", IEEE/ACM Transactions on Networking, Vol. 3 No. 4, pp. 365-386, August 1995.
- [Gil60] E.N. Gilbert, "*Capacity of a Burst-Noise Channel*", The Bell System Technical Journal, September 1960.
- [H261] ITU-T Recommendation H.261, "*Video Codec For Audiovisual Systems and Terminal Equipment*", March 1996.
- [JDS95] S. Jamin, P. B. Danzig, S. J. Shenker, L. Zhang, "*A Measurement based Admission control Algorithm for Integrated Services Packet Networks*", Proc. SIGCOMM'95, 1995.
- [MJV96] S. McCanne, V. Jacobson, M. Verletti, "*Receiver-driven layered multicast*", ACM Sigcomm'96, pages 117-130, 1996.
- [MCC95] S. McCanne, V. Jacobson, "*Vic: A Flexible Framework for Packet Video*", Proc. ACM Multimedia'95, San Francisco, CA, November 1995.
- [MeS99] J. Meggers, M. Schuba, "*Analysis of Feedback Error Control Schemes for Block Based Video Communication*", International Packet Video Workshop (PV' 99), New York City, April 1999
- [RHE99] R. Rejaie, M. Handley, D. Estrin, "*RAP: An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet*", to appear in IEEE INFOCOMM 99.
- [Sch96] H. Schulzrinne, "*RTP A Transport Protocol for Real-Time Applications*", RFC 1889, January 1996.
- [Sis97] Dorgham Sisalem, "*End-To-End Quality of Service Control Using Adaptive Applications*", in IFIP Fifth International Workshop on Quality of Service (IWQOS '97) , (New York, USA), IFIP WG 6.1, May 1997.
- [Sis98] Dorgham Sisalem, "*Fairness of Adaptive Multimedia Applications*" in the proceedings of ICC'98, June, Atlanta 1998.
- [SRC84] H. Saltzer, D.P. Reed, D. Clark, "*End-to-end arguments in system design*", ACM Transactions on Computing Systems, vol 2, no. 4, 1984.

- [TAL97] A. K. Talukdar, B. Badrinath, and A. Acharya, "*On accommodating Mobile Hosts in an Integrated Service Packet Network*", IEEE INFOCOM' 97, April 1997.
- [TSS97] David L. Tennenhouse, Jonathan M. Smith, W. David Sincoskie, David J. Wetherall, and Gary J. Minden, "*A Survey of Active Network Research*", IEEE Communications Magazine, Vol. 35, No. 1, pp 80-86. January 1997.
- [TuH96] T. Turletti, C. Huitema, "*RTP Payload Format for H.261 Video Streams*", RFC 2032, October 1996.
- [TUR96] T. Turletti, C. Huitema, "*IVS Videoconferencing in the Net*", IEEE ACM Transactions on Networking, Vol. 4, pp-340-351, June 1996.
- [WiZ98] R. Wittmann, M. Zitterbart, "*Amnet: Active Multicasting Network*", Proc. of Intern. Conf. on Communications (ICC'98), Atlanta, GA, USA, June 1998. IEEE.

INTERNET BASED SERVICE DEVELOPMENT

Olli Martikainen, Jouni Karvo

*Helsinki University of Technology,
Otakaari 1, FIN-02150 Espoo, FINLAND,
Tel. +358 9 451 2174, Telefax +358 9 451 5014,
E-mail {Olli.Martikainen, Jouni.Karvo}@hut.fi*

Abstract: The development of telecommunications services is affected by the changes in telecommunications market and in the service provisioning technologies. The market changes follow partly from the regulatory changes and the privatisation of the teleoperator industry and partly from the growing new service providers in mobile and Internet areas. This paper suggests that the current service creation architecture used by incumbent telecom operators may lose its significance in the future. Instead, new service architectures based on Internet and mobile service provisioning models will enter.

We present forecasts on Internet access and service market and evaluate the TCP/IP based traffic in future networks. The growth of Internet based services suggests a major change in future telecommunications service provisioning technologies. Furthermore, we present possible service platforms and service management environments available in future telecom service provisioning and conclude some major technology discontinuities in the service development.

1. INTRODUCTION

Telecommunications growth in Europe will in the next few years concentrate in the area of mobile- and data traffic. The turnover of services is depicted in Figure 1 (see [1] and [3]). It can be concluded from market research that the IP traffic already present in the current telecommunications networks will continue its increase. The amount of Internet traffic will outweigh the amount of traditional telephone traffic, and new services and

service architectures will evolve. The transition from the existing telecommunications services to Internet based ones will change the service infrastructure as well as service creation and management structures.

In Section 2 we describe how traffic volumes are expected to grow in the near future. Section 3 describes the service provisioning and management architectures, and the conclusions will be drawn in Section 4.

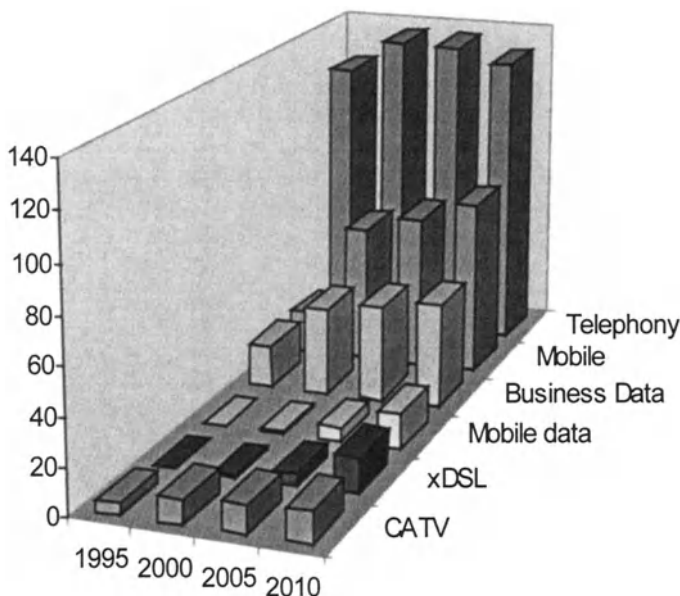


Figure 1. Telecom service turnover in Europe (Billion USD)

2. SERVICES

The incumbent teleoperators and their networks have been built on ISDN and B-ISDN switching where services are based on mobile, voice and IN technologies. The basic assumption has been that the peripherals are not intelligent, and all intelligence is located in network nodes. The Internet is based on the assumption on intelligent peripherals and less-intelligent network routers. The Internet growth, boosted by the invention of World-Wide Web in the early 90's, is expected to form the majority of traffic in the future telecommunications networks.

As estimated from the market studies of EITO [1] and OVUM [3] the expected Internet capable telecommunications terminal volumes n_i in

Europe are shown in Figure 2. In the figure NB and GPRS (Global Packet Radio System) mean narrowband fixed and mobile data, and xDSL and UMTS (Universal Mobile Telecommunications System) broadband fixed and mobile data, respectively. PDA (Personal Digital Assistant) can be fixed or mobile. The expected amount of traffic generated by the terminals is yielded by the weighted sum of the expected traffic and service time per day and terminal. Let d_i denote the traffic generated by a terminal of type i , t_i the expected service time per day of type i terminal and n_i the expected number of terminals of type i . Now, the total amount of traffic L in the network is given by the sum over all terminal types

$$L = \sum_i d_i t_i n_i.$$

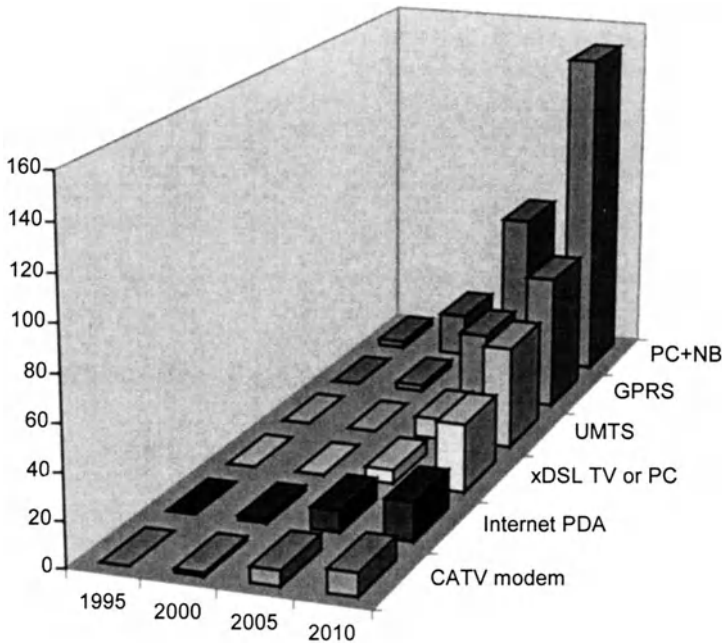


Figure 2. Internet terminals in Europe (million)

We divide services into two classes: Class 1 services could use the traditional circuit switched network with its service structure. These services include telephone, mobile phone and ISDN services. Class 2 services need seamless integration to the Internet world, namely the TCP/IP protocol suite and its service model. Typical class 2 services include electronic commerce, audiovisual content distribution and electronic

publishing. Class 2 service turnovers in Europe are estimated in Figure 3 by us using data from [1], [2] and [3].

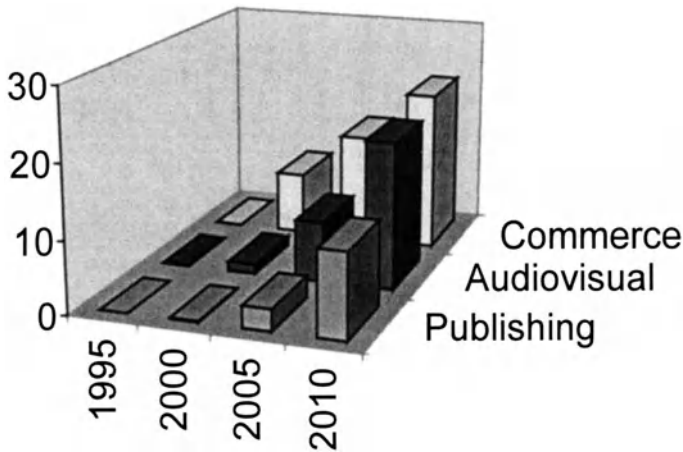


Figure 3. Turnover of TCP/IP based services in Europe (Billion USD)

Class 2 services will be based on ISDN, GPRS, xDSL, UMTS and interactive CATV connections with TCP/IP based services and Web based radio and TV programs. The expected parameters used in our calculation for each terminal type are the following:

Class 1:

- telephone and ISDN $d_i = 64$ kbps, $t_i = 10$ min/day
- mobile phone and GPRS $d_i = 64$ kbps, $t_i = 3$ min/day

Class 2:

- Web over ISDN $d_i = 64$ kbps, $t_i = 30$ min/day
- GPRS and Internet PDA $d_i = 64$ kbps, $t_i = 15$ min/day
- xDSL $d_i = 2$ Mbps, $t_i = 30$ min/day
- UMTS $d_i = 0.5$ Mbps, $t_i = 15$ min/day
- interact. CATV modem $d_i = 2$ Mbps, $t_i = 30$ min/day

For Class 1 we expect 243 million telephone and 111 million mobile connections in Europe in year 2000 [1]. Figure 4 shows the total expected amount of traffic L calculated separately for both classes of services. In the figure it can be seen that Internet type traffic will dominate after four years time. Since IP traffic dominates, service platforms and management should be developed based on Internet and Web models. In what follows we shall

compare traditional telecom service architecture with Internet and Web based service architectures.

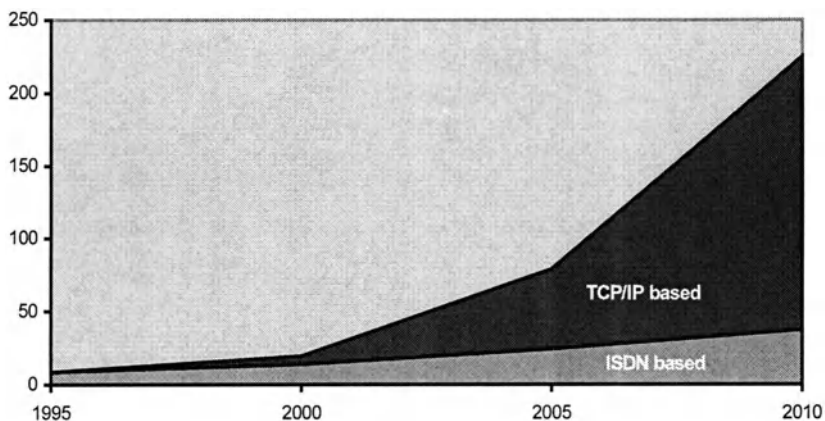


Figure 4. ISDN based and TCP/IP based traffic in Europe (Petabit/day)

3. SERVICE ARCHITECTURES

The incumbent teleoperator services are based on digital switches and service logic software developed for them. This is a centralized service model (Figure 5).

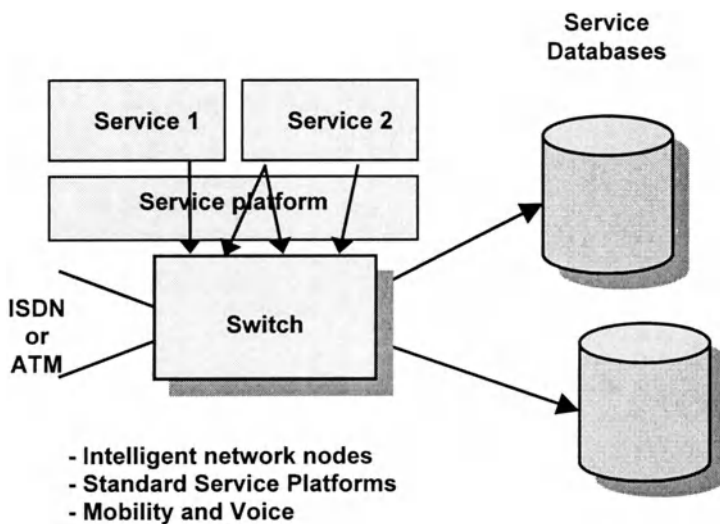


Figure 5. Traditional service model

Internet services are based on routers and the client-server model, where services are distributed in Web servers and accessible using the corresponding Uniform Resource Locators (URL) (Figure 6).

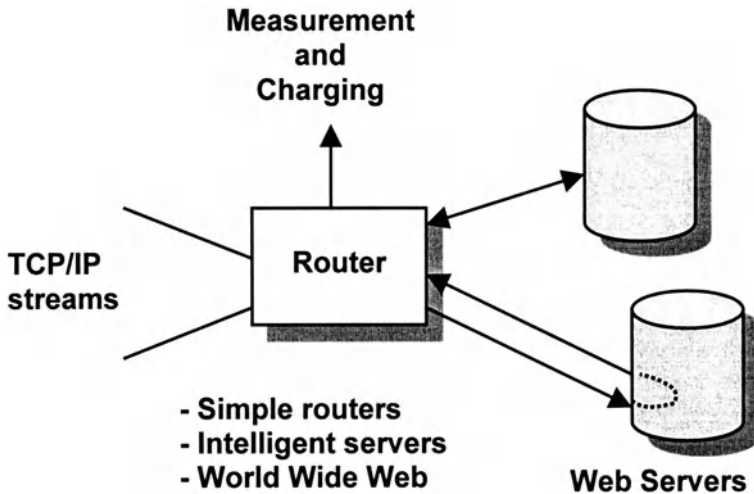


Figure 1. Internet and Web service model

Service management is based on service creation, parametrization and distribution in the network, customer identification and authentication, charging and customer billing, and necessary security features.

Incumbent teleoperators have customer databases (CDB) for customer identification and billing databases (BDB) for billing information. Customer authentication and billing is based on subscriber number (A-id) and security on closed signalling system (Figure 7). Service software is loaded in digital switches (DX) as centralized services, such as GSM and IN services. It is typical for incumbent teleoperators that the cost of the service management environment may be higher than the cost of service software and platforms.

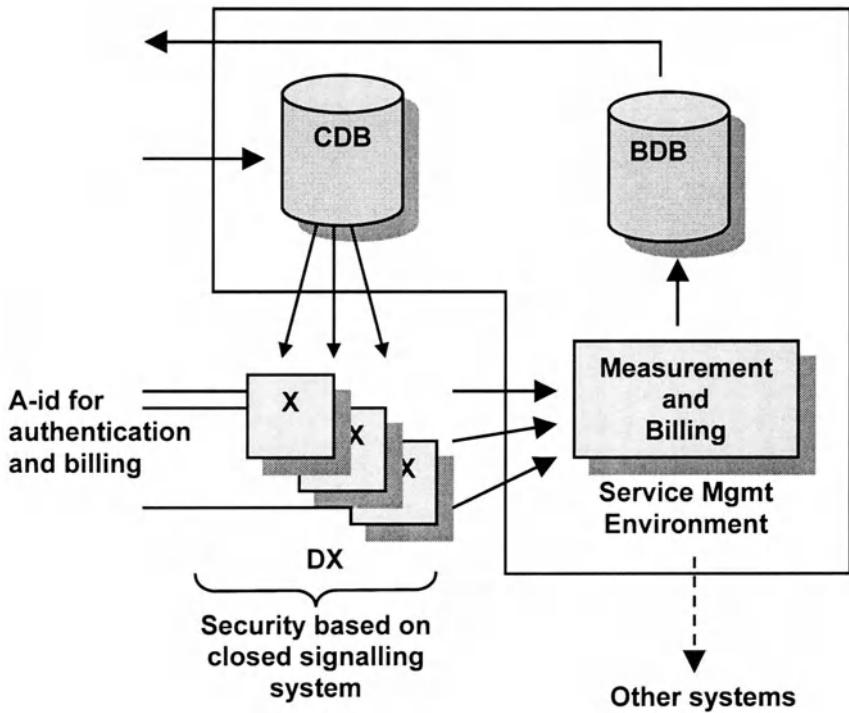


Figure 7. Traditional Service Management

Service management in Internet differs strongly from traditional service management. Users are not necessarily tied to any subscriber number so that user identification and authentication must be built in servers using specific applications. Service software is located in servers which are not usually owned by the teleoperators. Hence also the charging and customer billing must be built as separate applications. Moreover, network signalling and control data is carried on the same connections as the user and service specific data. This means that separate security functions must be built to support confidentiality in the network. As a result, the Internet service management relies on separate applications specified for the service management functions, which create a “service management layer” around routers (R) and basic TCP/IP connections (Figure 8).

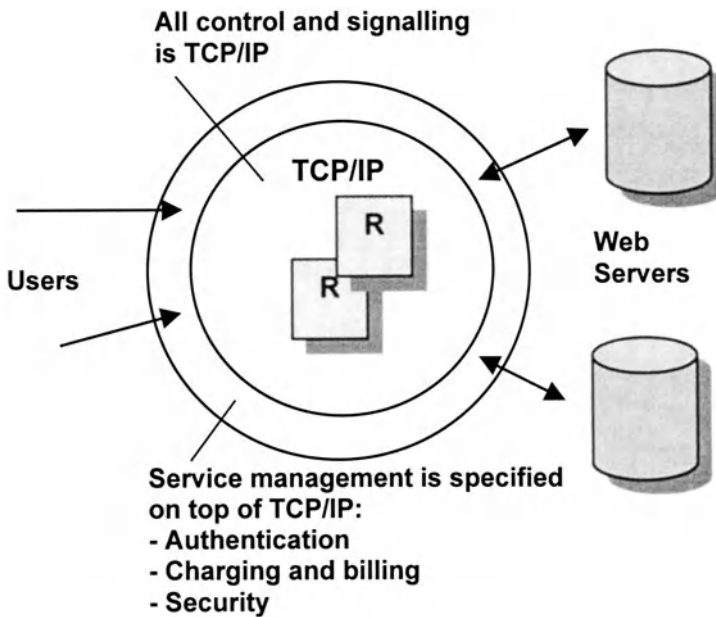


Figure 8. Internet Service Management

4. CONCLUSION

The service management environment takes care of service creation and distribution in the network and of customer and service related data. The difference of Internet based services compared to incumbent teleoperator services is not only in the service application structure and service platforms but also in service management environments.

In fact, differences in service management between traditional and Internet environments are more profound than differences in basic service applications. As a result we conclude that the new structure for service management needed in the Internet environment will be a key success factor for large scale TCP/IP based service provisioning in the near future.

REFERENCES

- [1] European Information Technology Observatory 98 (EITO) – European Economic Interest Grouping, <http://www.fvit-eurobit.de/eito>
- [2] IDATE 1998, <http://www.idate.fr>
- [3] OVUM Forecasts: Key Telecom Markets, 1998, <http://www.ovum.com>

SMART VS. INTELLIGENT VS. ACTIVE IP NETWORKS

Guy Pujolle

PRiSM - University of Versailles

45, avenue des Etats-Unis - 78000 Versailles – France

email: Guy.Pujolle@prism.uvsq.fr

Abstract: IP networks have to be controlled for providing quality of service necessary for multimedia services. Three possibilities could be proposed to reach this goal:

1. The parameters of the IP architecture are controlled through a multi-agent system. In other words a meta-control is applied to the different algorithms running over the network.
2. A second possibility is to copy what was performed in telecommunication networks: a very general architecture named Intelligent Network (IN). An implementation could be seen through the TINA architecture. However, the IP Intelligent Network platform could be Java/Jini-oriented and in some sense more distributed.
3. A third possibility would deal with active networks. In this case, different possibilities are also available that will be presented. As a conclusion, the three possibilities will be compared.

1. INTRODUCTION

In future IP networks one of the goals is to provide a large variety of services. For almost all services the networks will need to provide explicit Quality of Service (QoS) guarantees to subscribers. Several solutions are proposed by IETF through Integrated services, Differentiated services, other flow control and signaling solutions like RSVP. However, these schemes are very difficult to optimize and do not take care of the interface between the network and the application. An intermediate level that could be named "intelligent interface" has to be provided between upper and lower layers.

This software has to know what the application wants and how to control parameters of the different quality of service schemes implemented above. As a summary, one of the goals of introducing intelligence in a network is to optimize the resource but guaranteeing the service. Indeed, the intelligence to be added in the network should promote a rapid introduction of any new service with its quality of service through a flexible network architecture. This flexibility should be reached using new interfaces. Three possibilities are proposed in this paper:

- Use intelligence coming from some software agents that we could name "intelligent agents";
- Use a plate-form between the IP network and the application that looks like the Intelligent Network (IN) proposed by the telcos;
- Use the new paradigm of active networks.

The goal of this paper is to introduce these three schemes and give some comparisons.

The paper is composed as follows: intelligent agents will be introduced in section 2. Then, in section 3 we will describe the plate-form that could be developed over an IP network to control the interface between networking elements and application entities. Section 4 provides an overview of active networks and the way these networks could help the optimization of IP networks. Finally, in section 5 we try to compare the different solutions described in previous sections to play the role of this intelligent interface necessary to control networks and applications.

2. INTELLIGENT AGENT SYSTEMS

A first possibility to introduce intelligence in an IP network is to implement a distributed intelligent agent system inside the different routers and terminal equipment. The goal of this distributed system is to control the application and protocol parameters and so forth to optimize resource of the network. Distributed intelligent agent systems are good candidates for this task.

First of all, let us have a look to the definition of an intelligent agent before to have them grouped to perform a function. Indeed, an intelligent agent offers a variety of definitions:

- "The term agent is used to represent two orthogonal concepts. The first is the agent's ability for autonomous execution. The second is the agent's ability to perform domain oriented reasoning" [1].
- "An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors" [2].

- "Agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed" [3].
- "An agent is a persistent software entity dedicated to a specific purpose" [4].
- "Intelligent agents continuously perform three functions: perception of dynamic conditions in the environment, action to affect conditions in the environment, and reasoning to interpret perceptions, solve problems, draw inferences, and determine actions" [5].
- And so on...

Now the problem is to group these agents for providing a global task on a distributed system. The Distributed Artificial Intelligence (DAI) systems may be seen as such a grouping. The DAI field is often divided into two sub-areas: Distributed Problem-Solving (DPS) and Multi-Agent Systems (MAS). DPS considers how the work of solving a particular problem can be divided among a number of modules that cooperate by dividing and sharing knowledge about the problem and the developing solution. Multi-agent systems include intelligent behavior among a collection of possibly preexisting, autonomous, intelligent agents, when they can coordinate their knowledge, goals, skills, and plans to take action or solve problems [6]. The agents in a multi-agent system may be working toward a single global goal or separate individual but interacting goals [7].

Agents in MAS and DPS systems must share the knowledge about problems they face and solutions they reach, but they must also reason about the coordination processes among the agents.

DAI can be useful in domains in which action, perception, and/or control are naturally distributed. It is also a way to reproduce the human behavior when a group of experts works together to realize a task. For these reasons, multi-agent systems are envisaged to control an IP network. Several simple examples have been provided in the literature, using this kind of distributed intelligent agent systems. These systems permit a meta-control structure [8] [9]. Assuming that for each network node we associate one or several agents, the network can be seen as a multi-agent system in which the main goal is to ensure a control of the network.

The quality of service control can be seen at two levels: the local and the global level. Several agents in each node may be defined for controlling the quality of service. Each agent has to perform a specific procedure, which is triggered according to the state of the node and to the QoS required by each connection. This constitutes a local level for the congestion control. Moreover, agents can periodically interact to exchange their knowledge and ask to other agents if they need information they do not have. This constitutes the global level.

Intelligent agents are able to acquire and to process information about situations that are "not here and not now", i.e., spatially and temporally remote. By doing so, an agent may have a chance to avoid future problems or at least to reduce the effects. These capabilities allow agents to adapt their behavior according to the traffic flows going through the node.

Finally, interruptible agents may be used. They are able to act intelligently and to respond in real time. This capability allows implementing complex mechanisms able to take care of real time decisions.

3. IP INTELLIGENT NETWORK PLATFORM

IP networks environment is in essence a distributed application. From this point of view, modern distributed software techniques should be applied to the development and deployment of service [10]. Distributed computing has evolved from the use of primitive messaging-passing mechanisms to Remote Procedure Call (RPC), thus providing an abstraction of the underlying communication system and location transparencies. Recently, Distributed Object Computing (DOC) adds to RPC an object-oriented architectural framework for client-server applications. The most widely adopted DOC framework is CORBA, which can be already found in several telecommunications systems, including intelligent networks, service provisioning centers, mobile communication networks, and operations systems.

The Common Object Request Broker Architecture (CORBA) [11, 12] is a standard developed by the Object Management Group (OMG), which provides a model for interactions between objects in a distributed environment. There are several international groups developing architectures for telecommunication services that use CORBA as the platform for supporting the deployment and interoperation of the control and management software. In general, they start from the Open Distributed Modeling (ODP) concepts [13], as it is the case for both the Open Distributed Management Architecture (ODMA) defined by ISO/IEC JTC1/SC21 [14], and the Telecommunications Information Networking Architecture defined by the TINA Consortium (TINA-C) [15].

TINA proposes architecture for the provision and deployment of global services in such open telecommunications market. This architecture starts with the specification of an *enterprise model*, which identifies the roles (e.g., service and network providers, service and network managers, subscribers, and various types of users) which take part into the service, and the relationships among them. These relationships can be more complex than manager-agent paradigm, as they can involve more than two parties, and

peer-to-peer relationships encompassing negotiation capabilities. The requirements to model these types of relationships appear when dealing with control and management of multiparty services in which different types of roles interact.

TINA plate-form, through specialized SIBs that we can name MSIB (Management SIB) and CSIB (Control SIB), is able to adapt the network as a function of the flows and the quality of service required by these flows.

4. ACTIVE NETWORKS

A node in a store-and-forward network examines the packet going through interpreting the different fields. This is the work of a fixed virtual machine that is able to understand the values indicated in the different fields. An active network is also a store-and-forward network but the nodes are composed of a virtual machine able to interpret a specific language. Some commands may arrive in the nodes and these orders are interpreted and perform a computation resulting in a control of the packet.

- A first problem is the arrival of these commands that either may be carried out by the packets themselves or by special packets sometimes named smart packets.
- A second problem is the language to express the command and the powerness of this language is one of the most important points. That why these networks are also called programmable networks.
- A third problem is the level of the interface the end user may command the nodes. This interface is named Application Programmable Interface (API) and is under standardization through the P1520 IEEE committee.
- A fourth problem concerns the granularity of the control. Each packet may carry a new command or just one packet may decide of the control for all other packets during a quite long time.

Let us introduce the different types of active networks (see also [16], [17], [18], [19])

Most of the early active network architectures follow the "active packets" approach, which is fundamentally characterized by the fact that the code is carried by the packets. The nodes are also active because they allow computations up to the application layer to take place, but no active code resides in them. Therefore, the reason for calling these technologies "active packets" technologies is that active code is carried by the packets either to be executed on the data of the same packet that carries the code, or to be executed in order to change the state or the behavior of the node. Programs must be completely self-contained, thus discounting the need for persistent

states in a router. Also, programs must fit entirely into one packet (so they cannot be more than 1 Kbyte long) and the packet should not be fragmented.

A special protocol called active network Encapsulation Protocol (ANEP) was developed for the DARPA active networks program to facilitate portability and interoperability among different active networks projects [20].

The Active IP Option [21] describes an extension to the IP options mechanism that supports the embedding of program fragments in datagrams and the evaluation of these fragments as they traverse the network. Indeed, passive packets are replaced by active capsules, which are miniature programs that are executed as they travel. These capsules can invoke predefined primitives that interact with the local node environment, and leave information behind in a node that they have visited. Subsequent capsules can carry code that depends on this information. The capsule approach is an "in-band" approach in the sense that capsules carry the code along with the data on which it operates.

In the active node approach, the packets do not carry the actual code, but instead carry some identifiers or references to predefined functions that reside in the active nodes. The packets are active in the sense that they decide which functions are going to be executed on their data, and they provide the parameters for these functions. However, the actual code resides in the active nodes, it is not carried by the packets. The motivation for such architecture comes from the active packet approach. This approach suffers from either performance-related problems because safety and security requirements are huge, or capability related problems because the only way to minimize the security and safety issues is by restricting the programs that are carried in packets [39, 40].

ANTS [37] is an active network toolkit where arbitrary new protocols are automatically deployed at both intermediate nodes and end systems by using mobile code techniques. The network is viewed as a distributed programming system. The architecture introduces three components/schemes: capsules, active nodes, and code distribution.

Capsules are replacements for a packet. Their function is to include a reference to the forwarding routine to be used to process the capsule at each active node. The protocols need to be executed within a restricted environment that limits their access to shared resources. Active nodes play this role. During the processing, active nodes are responsible for the integrity of the network and handle any errors that may arise.

A Java-based prototype of ANTS has been created. The security of the implementation lies in the Java system itself. The choice of Java has allowed the researchers to evolve their architecture quickly but at the cost of less control over resources usage and lower absolute performance.

5. COMPARISON

The intelligent agent paradigm proposes to treat the network as multiple agent-friendly environments and the agents as programmatic entities that could move if they are mobile agents from location to location, performing tasks for users. The similarity with active networks is obvious. Indeed, many of the active network architectures use mobile code techniques that are very close to mobile software agent technology. However, the idea of active networks is much more general. Active networks visualize the network as a collection of active nodes that can perform any computations, and a collection of active packets that carry code and are indeed programs. Under that viewpoint, a mobile agent may be regarded as a specific type of an active packet, and a mobile-agent-compatible node of traditional networks could be regarded as a specific type of an active node since the latter is secure and allows any kind of computations. A fundamental difference between the two ideas is that active networks use the concept of network layer processing whereas mobile agent systems run as application programs.

Now, the plate-form solution may be viewed as an infrastructure able to interact with the end user and network protocols. This plate-form may be built using the multi-agent systems paradigm and is a general piece of software where agents may be regarded as a specific way to realize the plate-form.

As a summary, the three possibilities to introduce intelligence are not really concurrent but complementary. The intelligent agent paradigm is a way to implement intelligent plate-form and active networks. Intelligent plate-form and active networks are two layers that could be assembled to form the interface between applications and the hardware architecture.

6. CONCLUSION

We propose three solutions to introduce some intelligence in the control of future IP high-speed networks with QoS guarantee schemes. Indeed, these three solutions are based on different concepts that could be superposed on an IP networks. A powerful solution would be the implementation of an intelligent plate-form (ODP, INCM, TINA, etc.) built with cooperative intelligent agents for controlling QoS guaranteeing processes and based on an active network controlled by the intelligent plate-form. This architecture would be based on the three concepts developed in this paper: intelligent agents, IN, and active networks.

REFERENCES

- [1] <http://www.crystaliz.com/logicware/mubot.html/>
- [2] S. J. Russel, and P. Norvig, *Artificial intelligence: a modern approach*, Prentice Hall, 1995.
- [3] P. Maes, Artificial life meets entertainment: life like autonomous agents, *Communications of the ACM*, 38, 11, pp 108-114, 1995.
- [4] D.C. Smith, A. Cypher, and J. Spohrer, *KidSim: programming agents without a programming language*, *Communications of the ACM*, 37, 7, pp. 55-67, 1994.
- [5] B. Hayes-Roth, *An architecture for adaptive intelligent systems*, *Artificial Intelligent Journal*, Special Issue on agents and interactivity, 72, pp329-365, 1995.
- [6] E. Werner, *The design of multi-agent systems*, *Decentralized A.I.-3*, Elsevier Science Publishers B.V., 1992.
- [7] A. H. Bond, and L. Gasser, eds., *Readings in Distributed Artificial Intelligence*, San Mateo, Calif.: Morgan Kaufmann, 1988.
- [8] D. Gaïti, and G Pujolle, *Performance management issues in ATM networks : traffic and congestion control*, *IEEE/ACM Transactions on Networking*, 4, 2, 1996.
- [9] D. Gaïti, and N. Boukhatem, *Cooperative congestion control schemes in ATM networks*, *IEEE Communications Magazine*, 34, 11, pp. 102-110, 1996.
- [10] J. Pavón, *Building telecommunications management applications with CORBA*, *IEEE Com Survey*, 2, 1, 1999.
- [11] OMG, *The Common Object Request Broker: Architecture and Specification*, Revision 2.2, Feb. 1998.
- [12] S. Vinoski, *CORBA: integrating diverse applications within distributed heterogeneous environments*, *IEEE Commun. Mag.*, 35, 2, pp. 46-55, 1997.
- [13] ITU-T Recommendation X.90x: *Basic Reference Model of Open Distributed Processing*, ISO/IEC JTC1/SC21/WG7, Feb. 1995.
- [14] Open Distributed Management Architecture, ISO/IEC JTC1/SC21 N9947, 2/1997.
- [15] F. Dupuy, G. Nilsson, and Y. Inoue, *The TINA Consortium: Toward Networking Telecommunications Information Services*, *IEEE Commun. Mag.*, 33, 11, pp. 78-83, 1995.
- [16] D. L. Tennenhouse et al., *A survey of active network research*, *IEEE Commun. Mag.*, 35, 1, 1997.
- [17] IEEE Com. Magazine, *Special issue on Active and Programmable Networks*, xxxx 1998, vol. zz, no. yy.
- [18] IEEE Network, *Special issue on Active and Programmable Networks*, *IEEE Network*, 12, 3, 1998.
- [19] K. Psounis, *Active Networks: Applications, Security, Safety, and Architectures*, *IEEE Com Survey*, 2, 1, 1999.
- [20] D. S. Alexander et al., *Active Network Encapsulation Protocol (ANEP)*, RFC Draft, July 1997.
- [21] D. J. Wetherall and D. L. Tennenhouse, *The ACTIVE_IP option*, 7th ACM SIGOPS European Workshop.
- [22] S. Bhattacharjee, K. L. Calvert, and E. W. Zequra, *Implementation of an Active Networking Architecture*, Technical Report, Georgia Institute of Technology, July 1996.
- [23] S. Bhattacharjee, K. L. Calvert, and E. W. Zequra, *An Architecture for Active Networking*, *Proc. IEEE INFOCOM '97*, April 1997.
- [24] D. J. Wetherall, J. V. Guttag, and D. L. Tennenhouse, *ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols*, *Proc. IEEE OPENARCH '98*, April 1998.

ATM SECURITY

A Survey of Research, Products, and Standards

Extended abstract of the mini-tutorial

Herbert Leitold

Institute for Applied Information Processing and Communications (IAIK)

Graz University of Technology

Inffeldgasse 16a, A-8010 Graz, Austria

Herbert.Leitold@iaik.at

Key words: ATM security tutorial, ATM Forum Security Specification, ATM security products

Abstract: With the increasing deployment of asynchronous transfer mode (ATM) in both local area network (LAN), and wide area network (WAN) environments the interest in secure means of transferring sensitive information over ATM increases. Given potential application fields requiring Quality of Services (QoS) provided by ATM, as diverse as transferring medical images, linking branch offices to company networks, or governmental networks transferring sensitive citizen information, security is not only a desired option, it is rather an indispensable necessity.

The weakness of solutions to security in ATM networks has been recognised early. Cryptography is the tool of choice to achieve secure communications and a couple of proposals appeared to integrate cryptographic functions into ATM. Products have been developed which at the beginning faced the uncertainty of missing standards frameworks. However, this issue has been made up, as the ATM Forum has recently approved the Security Specification 1.0 which is in fact a comprehensive standardisation effort building the buttress for interoperable products.

In this tutorial, the ATM security arena is surveyed. It first gives an explanatory overview of ATM as a technology and addresses the challenges specific to ATM when aiming to migrate ATM to a secure infrastructure. Basing on these challenges, promising approaches to achieve security in ATM networks are described. By assessing the specific advantages and disadvantages, a comparison of the approaches against the ATM Forum Security Specification is made. In its final part, the tutorial surveys products

that are already available on the market or are announced by manufacturers working on ATM security.

1. INTRODUCTION

Networks based on cell relaying techniques origin from research done in the late 1970's and early 1980's as a novel approach to efficiently transport speech and video information (see [1] for a short description of cell relay history). With the first commercially available asynchronous transfer mode (ATM) switches, ATM network adaptors, and ATM local area network (LAN) solutions in the early 1990s [2], ATM gained a substantial stimulus and attracted the broad attention of numerous universities and research labs. Within less than a decade, ATM matured to a established technology in both LAN, and WAN environments, mainly because of the high bandwidth offered and the Quality of Services (QoS) guarantees.

However, almost together with the first ATM pilot installations not only the prospect of novel broadband networking applications became apparent, there have also been first rumours and concerns about security issues, as for instance expressed in a talk by Henning Schulzrinne "*ATM: Dangerous at any speed?*" [3]. In fact, ATM is as vulnerable to threats, such as wiretapping, forgery, or spoofing, as other networking technologies. What to some extent distinguishes ATM from most of the other networking technologies is, that efforts to overcome security risks have been made prior to its broad deployment.

In this mini-tutorial, security in ATM networks is discussed. Therefore, section 2 sketches the basics of ATM. The protocol reference model, as well as the principle architecture of ATM networks are addressed. In section 3, approaches to ATM security are surveyed by discussing relevant publications. An overview of standards related to ATM security is given in section 4. A survey of ATM security products is given in section 5 and, finally, conclusions are drawn.

2. ATM BASICS

Cell relaying is based on the idea of small data units of a fixed size—so-called cells—which carry a connection identifier in its header. Spoken in simple terms, the connection identifier enables efficient switching of the cells to appropriate outputs by the means of lookup tables or content

addressable memory (CAM) techniques, whereas the fixed size enables efficient buffering algorithms within the switching elements.

To associate the connection identifier to communicating end systems signalling functions are invoked upon establishing a connection—termed switched virtual connection (SVC)—or the association is fixed upon prior agreement with the network service provider—termed permanent virtual connection (PVC).

ATM promises to support the QoS required by a variety of services, such as real-time voice and video or delay insensitive data communication. Therefore, a number of ATM adaptation layers (AALs) are defined that convert the application data to the ATM cell format by invoking convergence sublayer (CS) and segmentation and reassembly (SAR) functions.

An ATM cell carries a five octet header and a 48 octet payload. The connection identifier is embedded into the ATM cell header as 24 bit virtual path identifier, virtual connection identifier (VPI, VCI) pair at the user network interface (UNI). In addition, a 3 bit payload type identifier (PTI) defines the ATM cell type, a cell loss priority (CLP) bit marks those cells that are preferably to be discarded, if this is required in network congestion periods, and a header error checksum (HEC) is provided in order to avoid misrouted cells in the case of bit errors in the header.

Beside the service independence, a major advantage of ATM is that it is defined independent from the physical media. Currently, 155 Mbps OC3c or 622 Mbps OC12c signals represent commonly used LAN end system and backbone links, whereas 1.5 Mbps T1, 2 Mbps E1, 34 Mbps E3, or 45 Mbps T3 signals for broadband WAN accesses are common. To support this broad range of signals, a physical layer embeds ATM cells to the transmission frame by employing transmission convergence (TC) and physical media dependent (PMD) functions.

Above stated characteristics represent the ATM protocol reference model [4], where the AAL interfaces to the application utilising ATM, respectively to the ATM layer which itself interfaces to the physical layer. In addition to these layers, a control plane performs the signalling to establish virtual channels (VCs), a user plane is responsible for transmitting the user data, and, finally, a management plane defines the functions to keep the network operational.

Associating the protocol reference model with the ATM network architecture, end systems at the UNI convert the user data to ATM cells by employing the AAL and transmit the bitstream resulting from the physical layer functions. The network elements—termed ATM switches—implement the signalling AAL (S-AAL, control plane) to associate end system connection requests to a VPI, VCI pair. In the user plane, ATM switches

operate at the ATM layer that mainly performs VPI, VCI translation, as in the general case these values of the header are modified whilst an ATM cell passes an ATM switch.

So how does security fit in? Given above sketched principle functioning of ATM networks, a number of threads can be identified, where some threads and user's perspectives to secure ATM communication are, as follows: By fooling the signalling procedures of the network, an end system might masquerade which results in authentic signalling, entity authentication, and data origin authentication as basic requirements. If ATM cells carrying a legitimate VPI, VCI are injected to an ingress switch, the additional data fools the receiving end system—data integrity proofs are desirable. As the ATM cells pass the network in plain text, wiretapping is possible which asks for data confidentiality. Finally, restricting the access to certain network resources asks for access control.

Cryptography is the tool of choice to meet these security requirements. In the following section of the tutorial, an overview to the research community's approaches to ATM security is given.

3. KEY PUBLICATIONS ON ATM SECURITY

In this section of the tutorial, the essentials of key publications in the ATM security arena are described, respectively an overview to common and distinct approaches is given in this abstract. For a more detailed description, the reader is guided to [5], where Donglin Liang reports on ATM security and gives a comprehensive reference list.

The references on ATM security can be categorised into two classes: Firstly, from an ATM network and service provider's viewpoint, issues such as availability, secure management, survivability, fraud, and billing are the focus. Secondly, from an end user's point of view, the major interest is in data confidentiality, authentication, data integrity, and access control. In this tutorial, the main focus is on the end user's perspectives, although the interests overlap, as availability or evading fraud are obvious user perspectives, respectively user security services constitute marketable added value for network providers.

For end-to-end security services, a possible distinction between different proposals and approaches is the embedding into the ATM protocol model. A major user perspective is that a security solution shall not counteract the benefits offered by ATM. With reference to section 2, these are QoS guarantees for different service classes and applicability to various physical media. Taking this consideration into account, embedding of security services into the ATM layer is an obvious choice, as the ATM layer is

independent from the AALs representing the different service classes, as well as independent from the physical layer. Consequently, for encrypting user data, the protocol data unit (PDU) to encrypt is the ATM cell payload. This approach raises two general challenges: Firstly, a general one in broadband networks is the high bandwidth to be dealt with. This leads to symmetrical ciphers implemented in hardware, such as the data encryption standard (DES), IDEA, or FEAL. The second challenge is ATM specific: As ATM cells are statistically multiplexed between different VCs, the requirement of having unique keys for each user connection—termed per-VC encryption—requires to change session keys rapidly between ATM cells assigned to different VCs. This is termed a key-agile encryption unit and discussed by Stevenson, et.al. in [6] and [7], the latter in fact amongst the best known publications on ATM security. Beyond that publications, in [8] a CAM-based approach to key agility is discussed. In [9], algorithm-agile encryption is introduced, where the encryption unit not just switches between keys, it even enables different encryption algorithms to be applied to different VCs. Finally, the concept of a ATM confidentiality system for 10 Gbps OC-192 is discussed in [10].

Approaches to confidentiality systems that are embedded into the AAL are discussed by Chuang [11], Deng, et.al. [12], Cruickshank, et.al. [13], and Laurent, et.al. [14]. However, these approaches face the problem that it requires an implementation for each AAL, such as one for real-time AAL-1 and one for computer-based communication using AAL-3/4 or AAL-5. Thus, the state-of-the-art is represented by above described ATM layer embedding of security services.

In addition to data confidentiality, data integrity is of importance. Therefore, message authentication codes (MAC) are appended to the PDU, respectively digital signatures are used for integrity proofs of signalling messages. At the ATM layer, the PDU is of a fixed size, i.e., the 48 octet ATM cell payload. Adding a MAC is not possible in the general case. For data integrity, embedding the MAC to the AAL is advantageous, as AAL-5 and AAL-3/4 frames are of variable size. This has been proposed by [14] respectively by Cherukuri, et.al. in [15] and Forné and Melús in [16].

For session key exchange, symmetric algorithms, such as block ciphers in cipher block chaining (CBC) operational mode, but mainly asymmetrical algorithms, such as Rivest, Shamir, Adleman (RSA) or Diffie Hellman (DH), are used for robustness and key distribution considerations. For performance reasons, symmetrical algorithms are used for re-negotiating keys in most cases, even if the initial negotiation uses asymmetrical algorithms. The relevant publications propose to either include authentication and key exchange into the signalling sequence, or to employ an in-band channel in the user plane. These approaches are discussed in

above stated publications [7], [11], and [14]. For authentication employing the same communication channels, as the key exchange protocols (user plane and control plane), established standards on digital certificates—mainly X.509—are used, as discussed in [12], respectively by Shankaran and Varadharalan in [17] who also discuss access control issues.

An overview to the ATM security services addressed in this section is given by Peyravian and Tarman [18] who both have been engaged as founder and chairpersons in the ATM Forum Security Working Group. This leads to the standardisation arena, which is discussed in the following section.

4. ESTABLISHED STANDARDS

In this section of the tutorial, relevant ATM security standards are discussed. These standards result from the efforts made by the ATM Forum to define normative references to migrate ATM to a secure infrastructure and are sketched in the following subsections.

4.1 ATM Forum Security Framework 1.0

In the Security Framework 1.0 [19], the ATM Forum defined ATM security objectives, identified threads, and derived general security requirements and services for ATM networks. This resulted in ten guiding principles, as follows (citations taken from [19]):

- AF SEC-1: The ATM network shall support capabilities to establish and verify the claimed identity of any actor in an ATM network (verification of identities).*
- AF SEC-2: The ATM network shall support capabilities to ensure that actors are prevented from gaining access to information resources they are not authorised to access (controlled access and authorisation).*
- AF SEC-3: The ATM network shall support the capability to keep stored and communicated data confidential (protection of confidentiality).*
- AF SEC-4: The ATM network shall support granting the integrity of stored and communicated data (protection of data integrity).*
- AF SEC-5: The ATM network shall support the capability that an entity can not deny the responsibility for any of its performed actions as well as their effects (strong accountability).*

- AF SEC-6: The ATM network shall support capability to retrieve information about security activities stored in the Network Elements with the possibility of tracing this information to individuals or entities (activity logging).*
- AF SEC-7: The ATM network shall support the capability to generate alarm notifications about certain adjustable and selective security related events (alarm reporting).*
- AF SEC-8: The ATM network shall support the capability to analyse and exploit logged data on security relevant events in order to check them on violations of system and network security (audit).*
- AF SEC-9: The ATM network shall support recovery from successful and attempted breaches on security (security recovery, management of security).*
- AF SEC-10: The ATM network shall support capabilities to manage the security services derived from the security requirements listed above (security recovery, management of security).*

In addition to above listed requirements, the ATM Forum Security Framework gives a plane specific interpretation of the ten security requirements. Basing on this framework, the ATM Forum Security Specification 1.0 [20] has been defined, which is sketched in the following subsection.

4.2 ATM Forum Security Specification 1.0

As a comprehensive standards effort to enable interoperable ATM security solutions, the ATM Forum Security Specification 1.0 [20] has been recently approved in its February 1999 version. It defines security services for the control plane, the user plane, and support services, as follows:

- **Control plane:** The specified security services are data origin authentication, data integrity, and the impact of error conditions such as incompatible signalling channel endpoints. Thus, the specification's scope is authentication and data integrity of ATM signalling in a hop-by-hop manner between adjacent signalling entities.
- **User plane:** For the user plane, entity authentication, confidentiality, data origin authentication, data integrity, and access control are defined. This results in a comprehensive normative reference to secure end-system communication.
- **Support services.** The support services define the protocols and for security information exchange, the key exchange functions, session key update protocols to renew negotiated session keys, respectively states

that X.509 shall be used for certificate exchange in the signalling channel or in-band in the user plane.

The specification defines a number of algorithm profiles for authentication, confidentiality, and data integrity. The cryptographic algorithm options employed are: DES, Triple-DES and FEAL in CBC or a counter operational mode are defined for data confidentiality. DES/CBC, FEAL/CBC digital signature algorithm (DSA), ESIGN, elliptic curve DSA-like, RSA signatures with secure hash algorithm (SHA) or message digest 5 (MD5) hash functions are employed for authentication and data integrity. For key exchange DES/CBC, TripleDES/CBC, FEAL/CBC, DH, RSA, and elliptic curve DH are defined, and finally, the message authentication codes that have been specified are DES/CBC, FEAL/CBC, MD5, and SHA.

4.3 UNI 4.0 and PNNI 1.0 Security Signaling Addendum

The ATM Forum Security Specification 1.0 defines additional signalling information elements (IEs) for carrying out support services in the control plane. These IEs are defined as a security addendum to the related signalling standards UNI 4.0 [21] and private network-network interface (PNNI) version 1.0 [22].

5. PRODUCT OVERVIEW

In its final section, the tutorial discusses different market available ATM security products, as well as recent product announcements. Although to some extent similar in its main characteristics and mainly oriented towards the ATM Forum Security Specification, each has interesting specific features which are outlined in the tutorial. The surveyed products are: Cyphercell™ [23] is a comprehensive implementation of the ATM Forum specification for 155 Mbps OC3c and currently under evaluation for the Common Criteria EAL4 security rating. HC-7910™ [24] is a key-agile encryptor that features a variety of physical interfaces and employs a proprietary encryption algorithm. In addition to being an ATM encryption unit, the ATLAS™ [25] series for OC3c has firewall functions employing a patented policy cache architecture. Again for OC3c, the Cylink ATM Encryptor [26] is a key agile unit. A National Security Agency (NSA) certified Type 1 encryptor capable of up to 622 Mbps OC12 is the unit FastLANE™ [27]. Featuring a number of physical interfaces up at 155 Mbps OC3c with a 622 Mbps OC12c

physical interface announced for fourth quarter of 1999 is the key agile encryption unit CellCase™ [27].

Above stated products have in common, that each is a line intercepting unit that is typically installed between an ATM end system or ATM switch and the ingress switch of the network service provider. The author of this paper is involved in the project Secure Communication in ATM Networks (SCAN) [28]. This project is embedded into the Advanced Communications and Services (ACTS) Programme of the European Commission. The SCAN objective is to bring ATM security services to the desktop by integrating a key agile high-speed ATM DES/TripleDES (HADES) unit to a 155 Mbps ATM network interface card for PCs.

6. CONCLUSIONS

In this paper the mini-tutorial on ATM security has been abstracted. In order to prepare for its major contents, the basics of ATM have been sketched and potential security threads have been identified. This has been followed by giving an overview to recent research related to ATM security and by outlining different approaches. Relevant standards that have been defined by the ATM Forum have been addressed and the extended abstract has been completed by giving a short survey of products that are established in the market.

REFERENCES

- [1] Cell relay frequently asked question, Who is the inventor of ATM?, subject G4 <http://cell-relay.indiana.edu/cell-relay/FAQ/ATM-FAQ/g/g4.htm> , 1997.
- [2] E. Biagioni., E. Cooper., R. Sansom, Designing a Practical ATM LAN, IEEE Network, v. 7, n. 2, 1993.
- [3] H. Schulzrinne, ATM: Dangerous at any Speed?, Proceedings of IEEE Gigabit Networking Workshop GBN'95, 1995.
- [4] ITU-T, B-ISDN Protocol Reference Model and its Application, Recommendation I.321, International Telecommunication Union, Telecommunication Standardisation Sector, 1991.
- [5] D: Liang, A Survey on ATM Security, student report, Ohio State University, Computer and Information Science Department, 1997. online available at: http://www.cis.ohio-state.edu/~jain/cis788-97/atm_security/
- [6] D. Stevenson, N. Hillery, G. Byrd, F. Gong, D. Winkelstein, Design of a Key Agile Cryptographic System for OC-12c Rate ATM, Proceedings of Internet Society Symposium on Network and Distributed Systems Security, 1995.
- [7] D. Stevenson, N. Hillery, and G. Byrd, Secure Communications in ATM Networks, Communications of the ACM, v. 38, n. 2, 1995.

- [8] H. Leitold, U. Payer, R. Posch, A Hardware Independent Encryption Model for ATM Devices, Proceedings of 14th Annual Computer Security Applications Conference ACSAC'98, 1998.
- [9] T.D. Tarman, R. L. Hutchinson, L. G. Pierson, P. E. Sholander, E. L. Witzke Algorithm-Agile Encryption in ATM Networks, IEEE Computer, v. 31, n. 9, 1998.
- [10] L.G. Pierson, E. L. Witzke, M. O. Bean, G. J. Trombley, Context Agile Encryption for High-Speed Communication Networks, ACM SIGCOMM, Computer Communications Review, v. 29, n. 1, 1999.
- [11] S.C. Chuang, Securing ATM Networks, Proceedings of 3rd ACM Conference on Computer and Communications Security, 1996.
- [12] R.H. Deng, L. Gong, A.A. Lazar, Securing Data Transfer in Asynchronous Transfer Mode Networks, Proceedings of Globecom'95, 1995.
- [13] H. Cruickshank, Z. Sun, S. Valentzas, A Proposal for Security Services in ATM Networks, Proceedings of the 4th IFIP Workshop on Performance Modelling and Evaluation of ATM Networks, 1996.
- [14] M. Laurent, O. Paul, P. Rolin, Securing communications over ATM Networks, in: Global IT Security, ed. L. Yngstrom, Chapman & Hall, 1997.
- [15] R.J. Cherukuri, M. Peyravian, S.F. Wu, A User Plane Security Protocol for ATM Networks, Proceedings of 5th International Conference on Telecommunication Systems, 1996.
- [16] J. Forné, J.L. Melús, An integrated solution for secure communications over B-ISDN, in: Communications and Multimedia Security II, ed. P. Horster, Chapman & Hall, 1996.
- [17] R. Shankaran, V. Varadharajan, Secure Signalling and Access Control for ATM Networks, Proceedings of 14th Annual Computer Security Applications Conference ACSAC'98, 1998.
- [18] M. Peyravian and T. Tarman, Asynchronous Transfer Mode Security, IEEE Network, v. 11, n. 3, 1997.
- [19] ATM Forum, ATM Security Framework 1.0, The ATM Forum Technical Committee, AF-SEC-0096.000, 1998.
- [20] ATM Forum, ATM Security Specification, Version 1.0, ATM Forum Technical Committee, ATM-SEC-01.0100, 1999.
- [21] ATM Forum, UNI Signaling 4.0 Security Addendum, ATM Forum Technical Committee, ATM-CS-0117.000, 1999.
- [22] ATM Forum, PNNI Version 1.0 Security Signaling Addendum, ATM Forum Technical Committee, ATM-CS-116.000, 1999.
- [23] CATM, CIPHERCELL™, online information at: <http://www.ctam.com.au/>
- [24] Cellware Broadband & Crypto AG; HC-7910 ATM Encryptor, online information at: <http://www.crypto.ch/>
- [25] StorageTek®, ATLAS 2002 and 2003 ATM Line Access and Security System, online information at: <http://www.stortek.com/StorageTek/network/NetSentry/ATLAS/>
- [26] Cylink, Cylink ATM Encryptor, online information at: <http://www.cylink.com/>
- [27] GTE, FASTLANE®, <http://www.gte.com/Products/Prods/fastlane2.html>
- [28] Celotek, CellCase, online information at: <http://www.celotek.com/>
- [29] SCAN, Secure Communication in ATM Networks, European Union, DXII funded ACTS project AC330, <http://www.iaik.tu-graz.ac.at/Research/SCAN/>

MOBILE AGENTS AND ACTIVE NETWORKS

Fritz Hohl

*Institute of Parallel and Distributed High-Performance Systems (IPVR),
University of Stuttgart, Germany*

Key words: Mobile agents, Mobile code, Active networks

1. INTRODUCTION

With the advent of mobile code, i.e. code that is sent to someone else's computer and executed there, a number of new technologies emerged that radically change some of the basic models of how networks and distributed applications work. Two of these new technologies are mobile agents and active networks. Mobile agents allow the programmer to employ "software robots" which wander around in the network, and fulfil tasks for a user. Active networks allow to add the ability to execute arbitrary code on routers and switches, thus gaining much more flexibility in the three lowest OSI layers.

In this tutorial, these two technologies will be presented and their relations exhibited. We start by defining the terms mobile code, mobile agent and active networks, and by disclosing the differences between them. Afterwards, we will examine the area of mobile agents and active networks in more detail.

2. MOBILE CODE, MOBILE AGENTS, AND ACTIVE NETWORKS

Mobile code allows to move code, i.e. procedures or even programs, to remote sites, and for executing the transferred code at these sites. Examples for mobile code systems are Java applets, remote evaluation (RPCs where the procedures are transferred before calling them) or even Postscript files (which are programs that are normally executed on printers).

Mobile Agents are program instances (or processes) capable of moving within the network under their own control. As a mobile agent contains code, mobile agents are special mobile code entities. The difference to mobile code lies in the inclusion of state (i.e. data state and execution state). This state is transported within the mobile agent, whereas mobile code entities are not able to transport state. However, the borders between both technologies are soft: if a mobile entity moves only once, it is hard to determine whether the entity is a mobile agent or just mobile code, especially if the entity is provided with some initial data. Even if the mobile entity travels along some nodes, we can think of this migration as a consecutive number of remote evaluations where the final state of one evaluation is (manually) taken as the parameters to the next evaluation. Apart from the technical characteristics, mobile agents also have another aspect: the usage as a programming paradigm. In this sense, mobile agents are a model where the programmer imagines to employ a breed of “software robots” wandering through the network, interacting with other agents and users. Note that the usage of this programming paradigm does not require a special technical realisation (just as object-oriented programming does not require object-oriented machine code), but an environment that allows the programmer to use this paradigm easily. Currently, the aspect of mobile agents as a paradigm is rarely addressed (see [BV99] and [GV97] for some exceptions).

An *active network* is a network where the transporting components (i.e. routers) are able to execute arbitrary code. This code is provided in some systems by special network packets (active packets) that can be injected by normal users. These active packets are of course mobile code entities. The difference to mobile agents lies on the one hand in the lack of transported state, on the other hand, active packets are executed in components that belong to the first three layers of the ISO OSI model, whereas mobile agents are layer seven entities (i.e. executed on end systems). However, the most radical view on active networks dissolves the OSI layering by arguing that active packets can be executed on every component of a network, be it a router, a switch, or an end system, thus allowing to dynamically place arbi-

trary functionality on any node. In this unified view, an active network is a more general model that builds the basis for mobile agents.

3. MOBILE AGENTS

Mobile agents are program instances that consist of three parts: code, data state, and execution state. They are able to migrate from one agent platform to another, taking all three parts with them. When it comes to migration, two types of agent systems can be distinguished: such that offer strong (or transparent) migration, and such that allow only for weak migration. Strong migration makes the migration process transparent to the agent programmer: the next statement after the migration command is executed on the target platform and the state of the program remains exactly the same as before, i.e. all three parts of the agent are transported automatically. Weak migration transports only code and data state of the agent, i.e. the execution state of the agent has to be en- and decoded manually by the agent programmer in case of a migration. Therefore, after a migration, often a start procedure is called.

In principle, mobile agents are a subtype of software agents, i.e. agents that have the ability to move. Practically the term software agent (or even agent) often denotes intelligent agents, i.e. agents that use AI techniques such as knowledge processing, although this usage is also an orthogonal attribute that may or may not apply to an agent.

Agent platforms play an important role for mobile agents, since agents can migrate only between them. These platforms protect the underlying computer system from attacks by the mobile agents on the one hand. On the other hand, agent platforms also execute the agents and provide functionality to them. This functionality includes:

communication

To communicate with other mobile agents, traditional mechanisms are offered by the systems, such as RPCs (or RMI), messages, tuple spaces etc. To address other agents, either a combination of a locally unique id plus the name of an agent platform are used (often for RPCs and messages) or communication elements (such as tuples) were specified. To cope with the problem of migrating communication partners, often either name services are used, or transparent proxy objects are employed that remain on a platform and relay communication requests to the target platform.

migration

If an agent requests a migration to another platform, the current platform stops the agent, encodes it into a transportable form, and sends it to the

target, often after having authenticated the target. More advanced version of migration ensure that the agent does not get lost even in case of node or communication failures during the migration.

Apart from these basic services, platforms sometimes offer other services such as persistency (i.e. storing data or agent snapshots), services lookup (i.e. finding agents that offer a service), locating agents (either locally or on a larger scale), integration of legacy systems such as databases, and the like. There is a larger number (currently over 60) of agent systems that have been designed by universities and industry (see [MAL] for an up-to-date and almost complete web listing of mobile agent systems). Most of them are research prototypes, only a few have been designed as a commercial product. Although nearly all of them use Java as the underlying basis, almost none of them is able to migrate mobile agents to another type of mobile agent system. There are standardization efforts (namely MASIF [MBB98] by the OMG and the FIPA proposals), but they do not specify migration interaction (in case of MASIF) or there were not implemented by agent systems (only 2 of them claim to implement MASIF, 1 implements FIPA).

Being the technical basis for applications, mobile agents offer a number of advantages. These include the *saving of network bandwidth* and *increasing of the overall performance* by allowing the application to process data on or near the source of data (e.g. a database), *asynchronous processing*, i.e. the possibility to fulfil a task without the need to have a permanent connection from the client to a host, *achieving true parallel computation* by employing a number of agents working on different nodes, *the replacement of a fault model* where network failures can interrupt in every phase of the computation by one where network failures can influence only the migration of an agent (as the rest is then done locally on the same node), and so on. Additionally, mobile agents “inherit” the advantages of mobile code systems, especially the possibility to transport functionality automatically to nodes where it has not been installed before.

Security is a very important aspect for mobile agent systems as both the system provider and the agent owner need to be sure that neither agents can attack the platform on a node nor that single platforms can attack agents, robbing e.g. transported digital money. Fortunately the one aspect, the protection of nodes against attacks by agents can be solved using techniques known from existing mobile code systems like Java applets. The other aspect, the protection of agents from attacks by the nodes, is not solved yet and constitutes a problem for the employment of mobile agent technology in open environments.

There are a number of application areas in which the usage of mobile agents is considered as being advantageous. These areas include among others: Information dissemination, mobile computing, information retrieval,

electronic commerce, software deployment and especially the management of telecommunication networks. Although mobile agents seems to offer many advantages and can be employed in some important application areas, currently, there is no broadly used real-world application that uses mobile agents.

The whole area of mobile agents was mainly inspired by the General Magic white papers [Whi94a],[Whi94b],[Whi95] where Jim White described Telescript, an early, but very mature commercial mobile agent system designed for electronic commerce applications. Another early paper that influenced mobile agent research very much is [HCK97]. If you are interested in the security aspects of mobile agents, [Vig98] can be recommended as a book that contains articles that represent some of the most important approaches in this area.

4. ACTIVE NETWORKS

Targeted at the lower layers of the OSI model (in contrast to mobile agents which work at the application layer), active networks try to overcome the shortcomings of traditional, passive networks: the difficulty of integrating new technologies into the existing network structure, poor performance due to redundant operations at several protocol layers, and the difficulty of accommodating new services in the existing architectural model. Additionally, applications have emerged recently that require computations within the network (e.g. firewalls, Web proxies, multicast routers). As no support for these application can be offered by the existing network infrastructure, computing power on regular, i.e. non-network nodes has to be allocated in order to fulfil these needs. In contrast to this, with active networks, the functions of a network component are no longer restricted by the built-in software of the vendors, but can be specified even dynamically by the applications that use the network.

Architectures for active networks can be distinguished according to the answer to the question of how activity is achieved. The first group of architectures, the “active packets” approaches, transport code in special packets, the second group, called the “active nodes” approaches place code in active nodes, allowing packets to carry code identifiers and parameters only. The third group uses flexible approaches that allows the user to choose between both approaches. Active packets approaches include the Smart Packets architecture (BBN Technologies) where a program has to be completely self-contained and has to fit into one unfragmented packet, thus allowing for only very compact programs, the Active IP Option (MIT) that allows to execute code fragments as an extension to the option field in the IP

protocol, and the M0 architecture (UCB, University of Zurich) which allows active packets to be very complex and powerful programs. Active nodes approaches include an architecture by the Georgia Institute of Technology where functions on active nodes are offered by network service providers, the DAN architecture (Washington University and ETH Zurich) that allows active nodes to load needed code from special code servers, and the ANTS architecture (MIT) where needed code is requested from neighbouring nodes. Approaches that combine both, active packets and active nodes include the SwitchWare project (University of Pennsylvania) which uses a layered architecture that allows the programmer to chose the trade-offs between the flexibility of the lower layer code execution and the speed and security of higher layers.

Security and safety are major concerns for active networks, as foreign programs were allowed to run on extremely security sensitive components. Breaches of security and even incorrect programs may result in a malfunctioning network, in privacy breaches, in attacks against other network parts to list just some possible threats. Some techniques that are used to achieve an adequate level of security are authentication, monitoring and control, limitation techniques, proof carrying code, as well as fault tolerance techniques and encryption. Authentication of active packets does not guarantee a program to be harmless, but allows to identify the author and the identity of a program, as well as the integrity of the code using public key techniques. Based on the identity of the program and other criteria, a reference monitor is then able to restrict the access of active packets to system resources and services. These restrictions may include also time and range limits (e.g. the number of hops). Proof carrying code allows programs to carry a proof with them that proves the correctness of the program given a specification.

There are a number of applications that profit from an architecture based on active networks. These applications include:

network management

Active networks can be used here to track and even to try to repair problems quickly without the need to maintain a communication connection from the management server to the managed components, which, in turn may be broken due to the problem it should handle. Further, the usage of active packets or nodes allows for tailoring management data and management policies.

multicasting

Using active networks for this application allows e.g. for dynamically moving multicast and repair packets caches to such routers that are “strategic” points, e.g. before wireless links etc., for suppressing negative

acknowledgements (NACKs) for originators that are known to be repaired in a short time, and for selectively sending repair packets to only those hosts which have requested it (which is a problem for multicast-like communication).

caching

With active networks, self-organizing caches are possible that determine dynamically where to place themselves depending on the current demand for data on different locations of the network.

See [Pso99] and [TSS98] for overviews of active network research.

5. LITERATURE

- [BV99] Bryce, Ciaran; Vitek, Jan: The JavaSeal Mobile Agent Kernel, in: Proceedings of the First International Symposium on Agent Systems and Applications (ASA '99) /Third International Symposium on Mobile Agents (MA '99) featuring the Third Dartmouth Workshop on Transportable Agents (DWTA '99). To appear.
- [CHK97] Chess, David; Harrison, Colin; Kershenbaum, Aaron: Mobile agents: Are they a good idea?. In Jan Vitek; Christian Tschudin (eds.): Mobile Object Systems: Towards the Programmable Internet, pages 25-45. Lecture Notes in Computer Science No. 1222. Springer-Verlag, April 1997.
<http://www.research.ibm.com/massive/mobag.ps>
- [GV97] Ghezzi, Carlo; Vigna, Giovanni; : Mobile Code Paradigms and Technologies: A Case Study, in: Rothermel; Popescu-Zeletin (eds.): Proceedings of the First International Workshop on Mobile Agents, MA'97, Springer-Verlag, 1997
- [MAL] The Mobile Agents List.
<http://www.informatik.uni-stuttgart.de/ipvr/vs/projekte/mole/mal/mal.html>
- [MBB98] Milojevic, D.; Breugst, M.; Busse, I.; Campbell, J.; Covaci, S.; et al: MASIF: The OMG Mobile Agent System Interoperability Facility, in: Rothermel; Hohl (eds.): Proceedings of the Second International Workshop on Mobile Agents, MA'98, Springer-Verlag, 1998
- [Pso99] Psounis, Konstantinos: Active Networks: Applications, Security, Safety, and Architectures, in: IEEE Communications Surveys, Vol. 2, No. 1, 1999
- [TSS98] Tennenhouse, David; Smith, Jonathan; Sincoskie, David; Wetherall, David; Minden, Gary: A Survey of Active Network Research, IEEE Commun. Mag., July 1998, vol. 36 no. 7, 1998
- [Vig98] Giovanni Vigna (Ed.): Mobile Agents and Security. Springer-Verlag, Germany, 1998
- [Whi94a] White, James: Telescript Technology: The Foundation for the Electronic Marketplace. General Magic White Paper, 1994
- [Whi94b] White, James: Telescript Technology: Scenes from the Electronic Marketplace. General Magic White Paper, 1994
- [Whi95] White, James: Telescript Technology: An Introduction of the Language. General Magic White Paper, 1995

DEPENDABILITY ISSUES IN SMART NETWORKS

Bjarne E. Helvik

Department of Telematics, NTNU

O. S. Bragstads Plass 2A

N-7491 Trondheim; Norway

E-mail: Bjarne.E.Helvik@item.ntnu.no

Key words: Dependability, availability, reliability, QoS, network survivability, fault-tolerance, logical faults, error persistence, error propagation, smart networks.

Abstract: The dependency of society and business on telecommunication services is commonly recognized, but is this conception taken into account in our effort towards smarter and more autonomous networks? The objective of the paper is to discuss some dependability issues in the context of these networks and to pinpoint some challenges. As an introduction, a brief review of dependability concepts is given. Next the following issues are discussed: a) strategies for providing a survivable transport network; b) fault-tolerant network nodes vs. fault-tolerant functionality on a distributed platform; c) software faults and their consequences like error propagation and network wide failure modes.

1 INTRODUCTION

Many foresee a future where we will have a “digital” existence in cyberspace that will be an integrated part of our private, social and professional life. Even if this future scenario does not become true, it is beyond doubt that our social and economic welfare is and will become increasingly more dependent on information and communication technology. However, when we pursue new technologies toward smarter networks, do we have in mind that we are going to thrust our welfare to the technology we are developing? Do we take into account that equipment may fail, that information may be deliberately or accidentally corrupted, that the environment may knock-out parts of the network, and that logical flaws in specification, design and implementation may cause instability and breakdown of an entire network?

History has shown that when failures may lead to disasters, the society tends to become conservative with respect to introducing new technology when a proven technology exists. In this situation, the technology deployment trend tends to be more towards marginal improvements than to introduce radically new solutions. The rather circumspect introduction of IN during the past decade may serve as an example. It should also be kept in mind that the additional cost of making a network dependable is similar to the cost of providing its basic functionality. To become a useful technology, smart networks must meet the above outlined conservatism and high cost, and be smart enough to deal with physical and logical faults in an economic and robust manner.

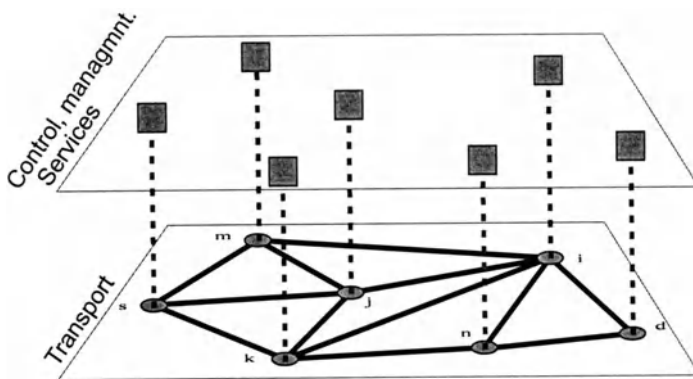


Figure 1. Simple two layer network model.

The objective of this paper is to introduce and discuss some issues concerning the dependability related to the coming networks. These challenges has been somewhat overlooked in the network research society. The dependability of the circuit switched communication infrastructure is taken for granted and up to recently, none has trusted their welfare on the Internet. Security issues, arising from intentionally man-made faults/intrusions, is recognized and put on the research agenda. Remember, however, that 'nature' causing unintentional physical and logical faults may be more inventive than man. This paper concentrates on these faults of the transport and control parts as the network as illustrated in Figure 1. In Section 3, the various options for making the transport part of the network tolerant to link and node failures are presented and trade-off with respect to speed and efficient use of spare capacity are discussed. How to provide fault tolerant service, control and management functionality are discussed in Section 4. Section 5 presents some observations of the effect of logical faults in network and points out this as a potential cause for major network outages. First, however, the basic dependability concepts are briefly revisited in the next section.

2 DEPENDABILITY

Dependability may be defined as *the trustworthiness of a system such that reliance can justifiably be placed on the service it delivers* [6]. A service delivered by a system is its behaviour, as it is perceived by its users. A user is another system (human or technical) which interacts with the former. For a more comprehensive introduction to dependability concepts, it is referred to [24]¹.

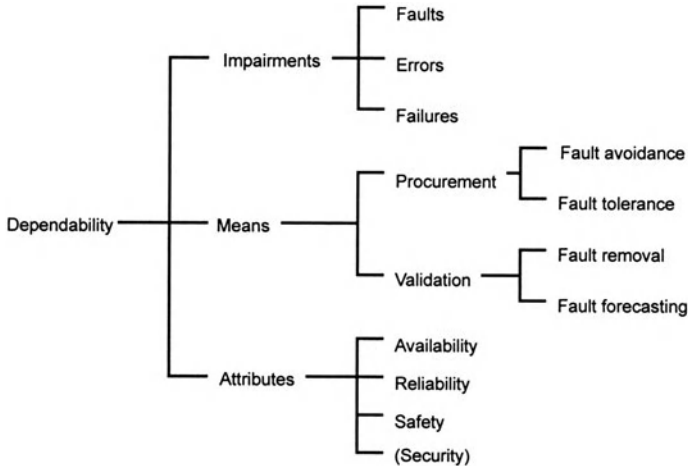


Figure 2. The dependability tree, linking various aspects of dependability [24].

2.1 Concepts

From an engineering point of view, we are left with the following questions: “What causes a system to be less than 100% trustworthy?”, “How should this, less than 100%, trustworthiness be quantified?” and “How should a required trustworthiness be achieved and assessed?” To handle these questions three classes of notions are introduced as illustrated in Figure 2:

- *Impairments* to dependability: faults, errors and failures, as well as their causes, consequences and characteristics. Impairments will be discussed in Section 2.2.
- *Means* to achieve reliability by fault avoidance and fault tolerance, see Section 2.3, and the means to validate and reach confidence in the result, like testing evaluation by modelling and mathematical analysis or simulation.

1. Other definitions exist. See for instance [16].

- *Attributes* of a system which describe the properties of a system with respect dependability, and whose quantification determine its dependability. The attributes may be regarded as the following abilities of a system:
Availability: to provide a set of services at a given instant of time or at any instant within a given time interval.
Reliability: to provide uninterrupted service.
Safety: to provide service without the occurrence of catastrophic failures.
Security: to prevent unauthorized access to and/or handling of information.

It is seen that the a quantification of the dependability attributes are significant QoS parameters of a system's service together with performance/traffic related parameters as for instance response time and delay.

2.2 Fault, errors and failures

Fault, error and failure are words which in daily language are used interchangeably and mean that something does not work, is incorrect, etc. However, in dependability engineering they denote separate phases from cause to consequence as illustrated in Figure 3. Starting with the consequence:

- *Failure*: delivery of incorrect service or the transition from correct service delivery to incorrect. A failure is a manifestation of an error that we observe on the outside of a system.
- *Error*: a system state that is liable to lead to failure, i.e. the manifestation of a fault within a system.
- *Fault*: the cause of an error. Note that faults may have a multitude of physical and human causes and be of various kinds as illustrated to the right in Figure 4.

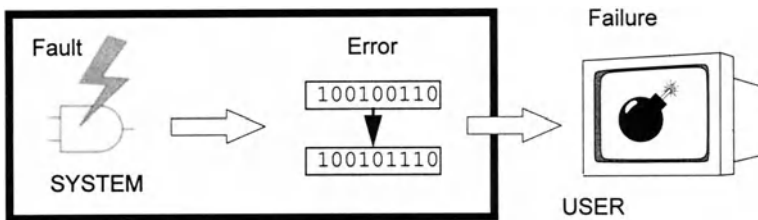


Figure 3. Relationship between faults, errors and failures.

This cause to consequence chain may be applied recursively with respect to both the development, deployment and operation, and the system structure. For instance: a misconception in a software developers head (fault) may lead to a wrong design (error) that results in the production of incorrect code (fail-

ure). This incorrect code represent a (dormant) fault in the system which when activated produces an error, which again may cause a failure of the system.

Often, only permanent hardware faults that are considered when the dependability of a system is regarded. This leads to a gross overestimation of the dependability, since these faults are the least frequent cause of failures. Human made logical (e.g. software) and operational faults as well as transient faults are far more frequent. See for instance [2, 10, 21].

2.3 Fault-tolerance

There are two basic approaches to obtain dependable systems, fault avoidance and fault tolerance. The basic principles are illustrated in Figure 4. In the fault avoidance approach, we seek to avoid the faults, e.g. by extensive quality assurance of the software, benign operating conditions for the hardware, etc. In the fault tolerant approach, the system is given a structure and mechanisms, which prevent errors in the system to manifest themselves as failures. A number of mature and well known techniques exist, e.g. FEC (forward error correcting codes) or CRC (cyclic redundancy check) combined with retransmission of errored packets to tolerate transient transmission faults. Other techniques is rerouting of traffic when network links and nodes fail, as introduced in Section 3 and replication of computing functionality as dealt with in Section 4. Introductions to fault tolerant design may for instance be found in [3, 17, 18, 26 Chap. 14, 28, 37].

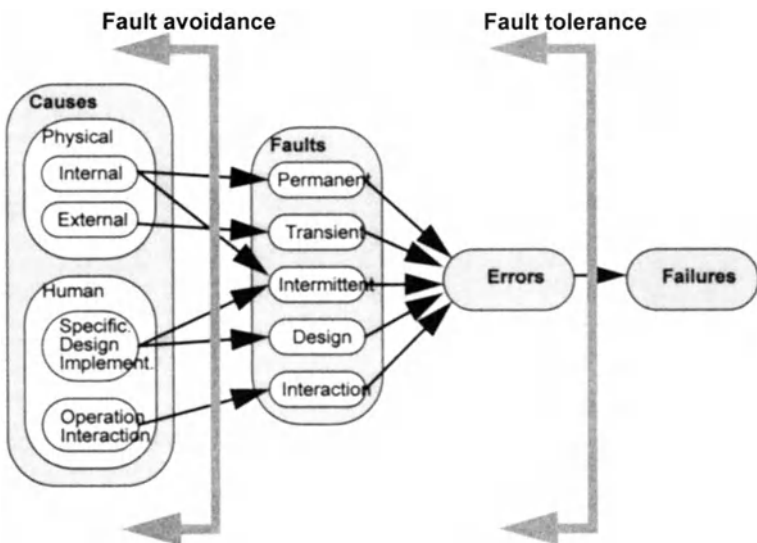


Figure 4. The design barriers of fault avoidance and tolerance to improve dependability.

Below, the options for fault-tolerance in smart network will be discussed. Neither fault avoidance nor fault tolerance comes cheap, and it is not straight forward to obtain a dependable network. Hence, smart networks should be inherently robust, i.e. fault tolerance of network services should be simple to achieve and do not require more additional development effort and deployed equipment than strictly necessary. Preferably smart networks should also minimize the deployment and operation efforts as pursued in the plug-and-play architecture [1].

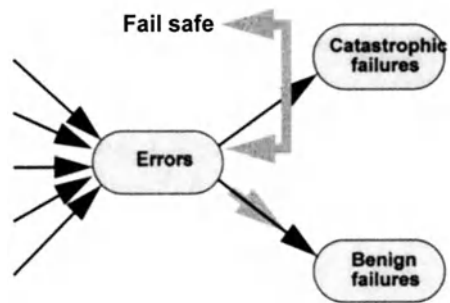


Figure 5. The fail-safe principle.

3 SURVIVABLE TRANSPORT

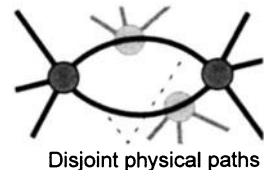
Various fault-tolerance strategies are employed to achieve dependable transport networks. There are three important issues in the choice of strategy:

- The redundant network elements and capacity needed, i.e. the additional equipment cost required to implement the scheme;
- The duration from the failure of a network element and until the transport service is restored;
- Centralized or decentralized control. Control by a dedicated management functionality or as an integrated part of the traffic handling.

In this section, three basic strategies, which differ with respect to the above listed issues are presented. Section 3.4 gives a brief discussion of these in the layered architecture and fault handling in transport networks.

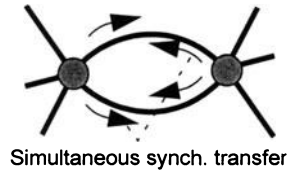
3.1 Protection

Protection is the simplest form of fault tolerance of the transport service between to nodes of a network, where a dedicated spare path is established. There may be intermediate nodes as indicated in the figure, but these do not perform any switching or rerouting of the protected traffic. The principle is, as illustrated in the figure, that the traffic between two nodes follows two (or more) dedicated paths. To have independent failing, the paths must be physically disjoint. In

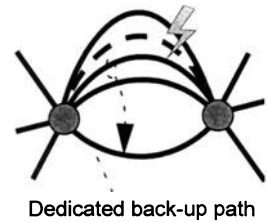


some cases this is sought achieved by minimum physical distance (e.g., 1.5 meters) of the cables combined with other means for physical separation. It is three types of protection switching:

- a) *1+1 protection*: The bit stream is sent synchronously on both paths toward the destination. The destination selects the bit stream it considers as the one with the higher integrity, i.e., fewer bit errors. It is ensured that path failures are detected by coding techniques, loss of signal, loss of synchronization, etc. The advantage of 1+1 protection is, as with all masking redundancy techniques, an immediate (in the order of a few ms) handling of path failures. A path failure should be unnoticed by the users of the transport service.



- b) *1:1* and *1:N protection*: are stand by protection strategies, where a dedicated spare (or back-up) path is available for one and N active paths respectively. Failure detection techniques are as for the masking redundancy, but the receiving node must inform the sending node which redirects the data stream. This incurs a temporary loss of service (in the order of 10 - 100 ms), but is usually tolerated by the end-user (applications).



In conclusion the restoration is simple, fast and proven, but requires more transmission capacity (but less costly control) than the (smarter) alternatives outlined below.

3.2 Reconfiguration

The reconfiguration strategy is based upon a centralized management of the network. A network management system, see Figure 6, supervises and controls all network resources. It seeks to keep a map of all network resources, their utilization and status. Based on this information, it sets up transmission paths between nodes in the network. When a network failure occurs, the network management system reconfigures the routing through the network.

This strategy has the potential to use the transmission capacity in the network very efficiently due to its ability to perform a global optimization. Reconfiguration may, however, be slow, in the order of minutes, due to the time needed for data collection, deciding upon a new configuration, and propagating the new configuration back to the switches. The strategy is vulnera-

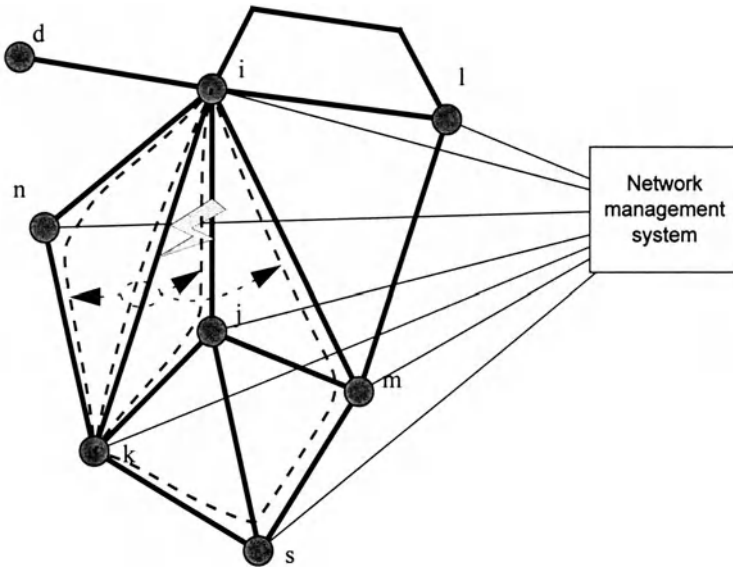


Figure 6. Reconfiguration of a network by centralized fault-handling.

ble, since it is both centralized and depends on information transfer in a network with failure(s) not yet handled. To overcome these drawbacks, parts of the management functionality are sought distributed, see for instance [7].

3.3 Self-healing

Self-healing is used as common nominator for a range of strategies which have in common that they have distributed control and requires no dedicated pre-reserved transmission capacity. The next subsections present three basic techniques. The two first are, as protection and reconfiguration, associated with transmission paths in the network, while rerouting in Section 3.3.3 is associated with the individual connections/streams.

3.3.1 Back-up paths

When a path (the active) is established through the network one or more back-up (stand-by) paths are established simultaneously. This/these back-up path(s) may be completely or partially disjoint from the active. A back up path does not carry any traffic unless a node or link along the active fails. Transmission resources along a back-up path may be assigned on-demand when an active path failure occurs. In this case, traffic may be lost if sufficient capacity are unavailable.

Hence, if the network shall meet dependability/QoS requirements, semi-dedicated resources must be reserved. See Figure 7 for an example. The link $[i, k]$ carries the active path between nodes i and k and has reserved stand-by capacity for either the active path $\{i, n\}$ or the active the path $\{i, m, s\}$. Since these paths have no common network element except the originating node i , failures of these two active paths are assumed independent.

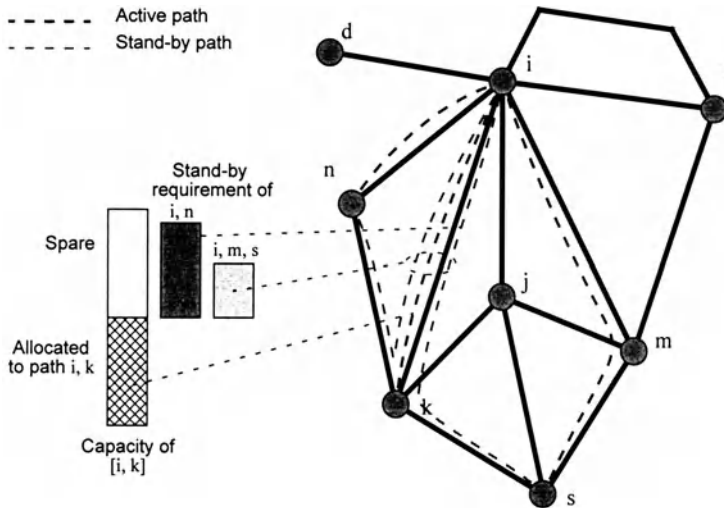


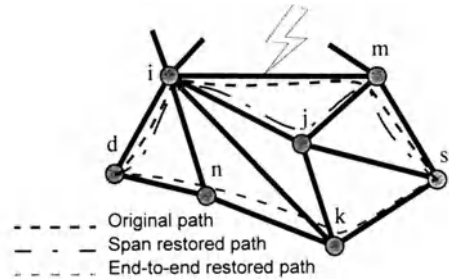
Figure 7. Example of allocation of back-up paths with semi-dedicated resources.

A path failure is detected in the end-nodes, for instance by missing OAM cells in an ATM connection. When resources on stand-by paths are available, the service may be restored in the order of 10 - 100 ms. A challenge is to allocate and adapt the back-up paths so the spare capacity in the network is optimally used under a given traffic load.

3.3.2 Flooding

Flooding is a fully distributed network management technique to restore communication after an element failure. The technique has got its name since it floods the network with request in the search for available capacity to replace that of a failed link. No resource reservation is made ahead of the reconfiguration, but the network must be dimensioned so that spare capacity is available. There are four variants of the method, dependent on whether:

- A new path restored by diverting the affected paths around the failed network elements failed, i.e., *span restoration*, or whether a new path is established between the source and destination node, i.e., *end-to-end restoration*. See the illustration to the right for examples.



- The search for a new (unidirectional) path starts from one node and terminates in the other, i.e., *single search*, or starts in both nodes and terminates when the searches “meet in the middle”, i.e., *double search*.

Flooding, especially end-to-end restoration, poses a number of algorithmic problems on how to reserve and release capacity to avoid deadlocks and to make an efficient use of the available resources in the network. The service restoration time is at least several seconds, and no dependability/QoS guarantees may be given, since no reservations are made.

3.3.3 Rerouting

This self-healing strategy is executed on the connection level. The traffic through the network finds new routes between source and destination end systems when a network element fails, i.e. by directing the packets (in connectionless communication) along new routes, or re-establishing the end-to-end connections along a new route. See Figure 8 for an illustration. Note that

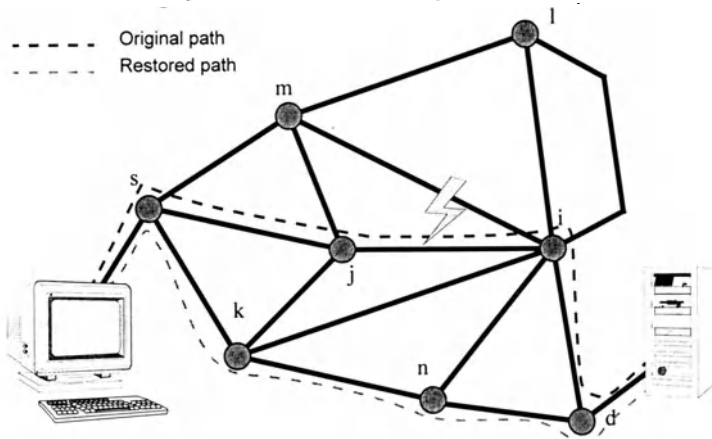


Figure 8. Example of self-healing in a network by rerouting connections or flows after a network element failure.

in this strategy, individual packets/connections are rerouted or re-established. This is different from the self healing strategies presented above, as well as protection and reconfiguration, which is aimed at restoring the end-to-end

paths of the network after an element failure. Each path may carry a number of individual end-user connections and/or messages between end-users. This path rerouting/re-establishment may be considered as a bulk restoration of the transport capacity by the network itself without involving the end-systems, and where it is an objective that the fault handling should be as “invisible” to the end-user (applications) as possible. Applying rerouting, this near transparency of network faults to the end-user (systems) will be lost, *no dependability/QoS guarantees can be made* and:

- the end-user application will stop, which may be critical for some applications, e.g., tele-medicine,
- longer time is needed before a service may be re-established after a failure, e.g., a new connection must be made or a period with large packet losses and/or delays will arise in connectionless networks,
- the network may be temporarily overloaded with connection requests or repeated packets after a failure, and
- semipermanent connections established by the management system, e.g., leased lines, require special attention.

There are also a number of advantages by utilizing this strategy:

- + less functionality is needed in the network since ordinary traffic handling functions are used for network element failures,
- + the routing algorithm used for rerouting after a network element failure, may be the same algorithm used to adapt to normal fluctuations of the traffic load, like daily variations and overloads,
- + the restoration is distributed (unless a centralized routing function is introduced) and hence, no centralized function forms a dependability bottleneck,
- + the path restorations are end-to-end and the information flows that shall be restored have a finer granularity, which yields a better resource utilization.

3.4 Multilevel fault handling

Table 1 summarizes some of the basic implementation issues related to the redundancy strategies discussed hereto in Section 3. Transport networks have typically a layered design, at least a physical layer, a transmission (link) layer and a switching layer. These layers may contain sublayers, e.g. the ATM virtual path and connection layers. In which of these layers should we introduce the fault tolerance? What fault-tolerance strategy should be used? Since the layers are designed independently of each other, how should we avoid introducing excessive spare capacity and avoiding that several layers compete to handle the same failure?

Table 1. Summary of resource handling issues for the various network redundancy strategies.

	Resources			Control		
	Dedicated	Pre-planned Reserved	On demand	Central	Distrib. control	Distrib. man-gemnt.
Protection	4					4
Reconfigura-tion		4	4	4		
Self healing		4	4		4	4

An exaggerated illustration of the problem is given in Figure 9. For instance, the cross-connect switching carried out by the SDH transport system, may alternatively be carried out by the ATM VP layer. If we are interested in IP transport, why do we not get rid of all the underlying layers and use IP-over-light? The answers to these questions are complicated by the fact that:

- in contradiction to what is illustrated in Figure 9, networks are not homogeneously and strictly layered end-to-end. For instance, an IP network may span several underlying networks;
- the lower layers may carry several independent transport services;
- all transport services (as well as end-users) do not have the same dependability requirements and hence the same need for fault-tolerance in the network.

For discussions on design of multilayer networks and on which layer that it is most profitable to introduce redundancy, see for [19, 20, 29 and 40]. Some rules of thumb may be given:

- a) The higher up in the network the larger is the flexibility and the number of stand-by resources needed is likely to be less;
- b) The higher up in the network, the larger is the delay from the failure event and until the service is restored;
- c) Low layer redundancy handling tends to introduce the least complexity since the mechanisms may be simpler and the redundancy handling may be common to a number of higher layer network services.
- d) Fault tolerance on a layer can only handle fault at its level and on the levels below. Hence, some fault tolerance is required at the highest level.

A closing question: with the observations of the current internet in mind [33, 22], is the currently so popular IP-over-light scenario able to shortcut today's multilevel fault handling and provide a fast enough reconfiguration and a stable rerouting in a large network where failures are the rule more than the exception?

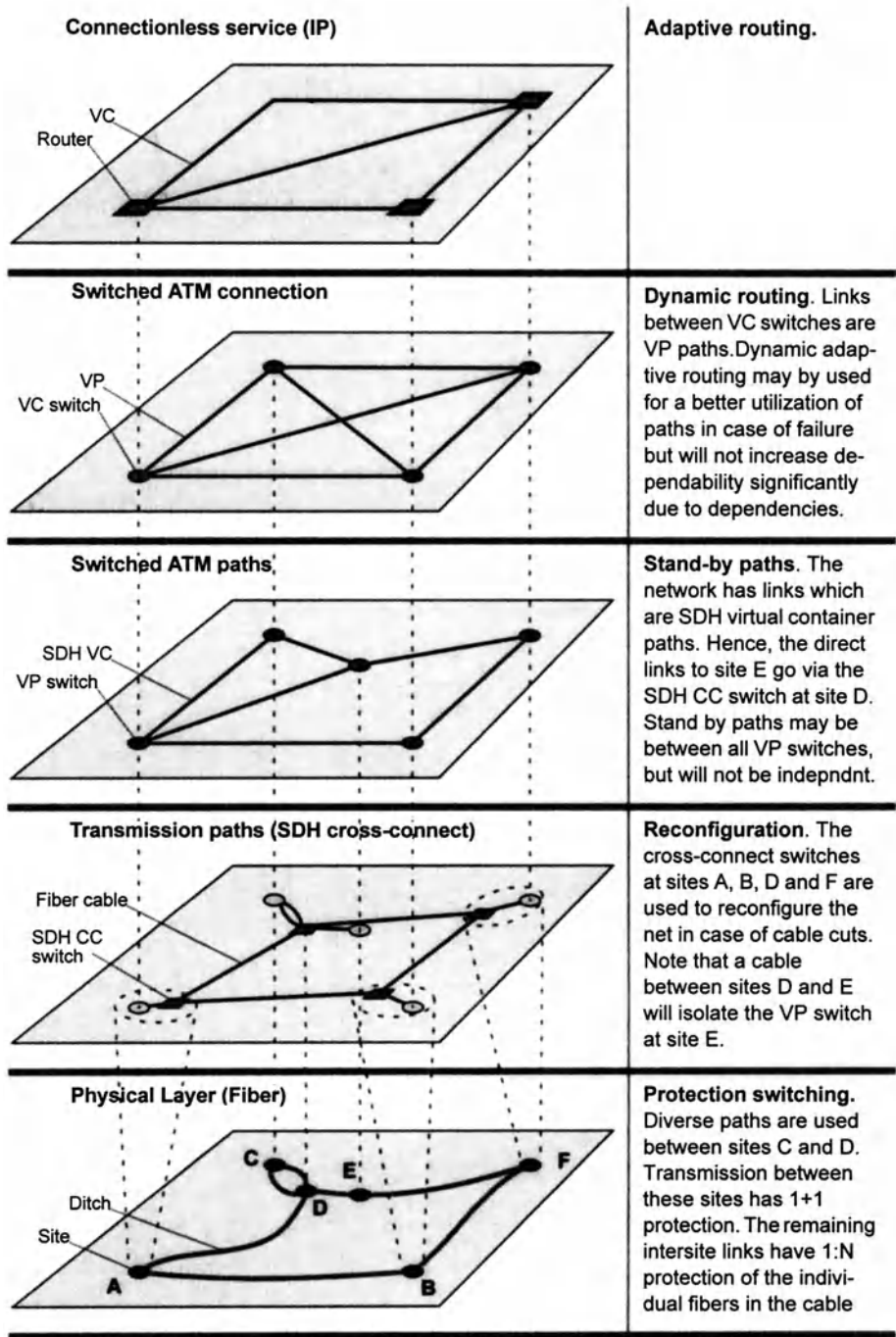


Figure 9. Layered network scenario with fault handling in multiple layers.

4 FAULT TOLERANT NETWORK FUNCTIONS

In the previous section, the techniques for making the transport services of a network fault tolerant was discussed. This section presents two basic fault-tolerance strategies that enable network functions to be performed, uninterrupted if required, in spite of computing and data storing element failures. Network functions in this context encompass call/connection/stream control, network and service management, and service handling as illustrated in Figure 1. The faults sought tolerated are primarily physical faults, even though some of these techniques also have been experienced to tolerate a subset of logical faults reasonably well [10].

The two strategies regarded for achieving fault tolerance in the computing and data storage functionality are:

- To use fault-tolerant network nodes, i.e. to make the computing and database platforms supporting the functionality of the network nodes fault-tolerant. For instance to make an intelligent network SCP fault tolerant by using a duplicated synchronous computer to execute the functions.
- To introduce fault tolerance at the network level, i.e. to have co-operating replicas of software objects/processes providing network functions in several nodes and thereby enable tolerance of node failures. Network level fault-tolerance is based on dependable distributed computing technologies.

The aim of the section is to view these as competing options for achieving dependability in smart networks. Hence, (too) little attention is paid to the “in-between” systems, e.g. [34, 30], which uses a specially designed transport network to ensure consistencies between replica.

4.1 Fault-tolerant network nodes

This strategy is illustrated in Figure 10, where each node in the network, and the services it delivers, are made fault tolerant by having redundant computing and/or storage capacity in the node. The replicas of the processes executed or stored are located within the same node. The architecture of the nodes may vary as illustrated. Fault-tolerance may for instance be achieved by micro-synchronous duplication, see node *s* of Figure 10, as in for instance the AXE [32]. Alternatively a distributed and mainly load shared architecture, see node *d* of Figure 10, as in Alcatel System 12 [4], may be applied. See for instance [3] for a discussion on the design and analysis of such systems.

As indicated by the type and age of the above references, the fault tolerant network node is the classic approach for public communication systems. This is a proven technology, and the design goal, originally set for the ESS 1 [8],

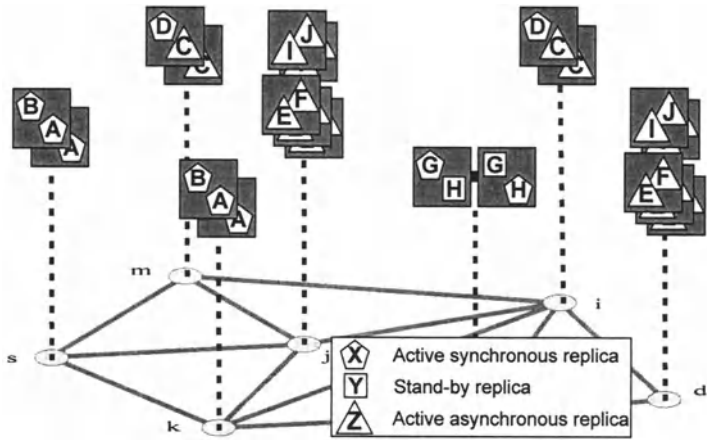


Figure 10. Illustration of network with fault tolerant nodes with various architectures.

of an expected down time of three minutes per year was achieved between one and two decades ago. In addition, this approach has several advantages. In most systems, it hides the fault-tolerance mechanisms for the application designer. The synchronization between replica incurs little or no delay, and the strict real time QoS requirements of communication systems is met. However, these systems are dedicated (legacy) systems and are more expensive than off-the-shelf computer and database systems. Located at a single site these systems are vulnerable to environment failures like fire.

Nevertheless, this is a viable approach to providing dependable network functions in a smart network, cf. for instance [15]. With respect to providing network functionality, the problems related to the handling of (physical) faults may be substantially reduced/hidden, but at a certain cost and with reduced applicability to smaller nodes.

4.2 Replication across the network

This strategy is illustrated in Figure 11. The computing and storing hardware of the nodes in the network are *not* fault tolerant. The services delivered by the network, however, may be fault tolerant by having replicas of the various service providing software objects/modules in several network nodes. The replication may be handled by appropriate middleware [27]. As in the strategy of Section 4.1, this requires redundant computing and storage capacity in the network. However, off-the-shelf equipment may be used and the redundancy installed for the various object/modules may be tailored to the dependability requirements of the services they provide and the required consistency between replicas. For instance, in Figure 11, there is three synchronous replicas of object/module A. These replicas have consistent states and by voting, an arbitrary failure of one of the replicas, as well as the node host-

ing it, may be tolerated. The two synchronous replicas of D and F allow a crash failure without interrupting the service. The active object/modules B and E have stand-by replicas that may take over if the active replica/node fails. The stand-by replica may have a consistent or partially consistent state with the primary, e.g. by checkpointing, or be in a default state. In the last case illustrated, C has three load-shared replicas without consistent states.

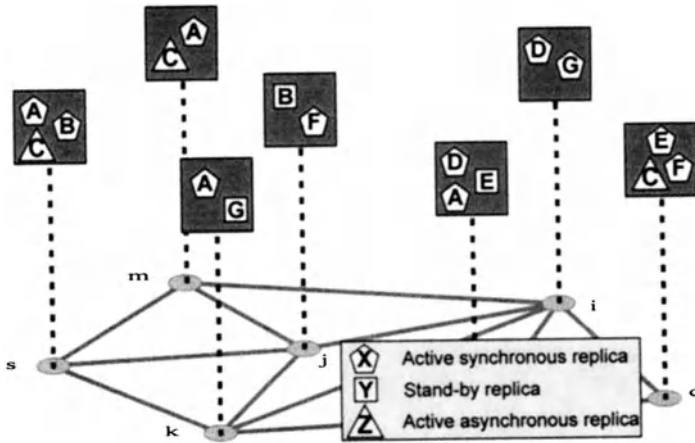


Figure 11. Illustration of network where software objects are replicated on disjoint network nodes for fault tolerance.

It is a common objective that replicas shall be location and fault transparent, i.e. users of the services shall not need to know where the replicas are located and whether some of the replicas has failed. Achieving a flexible replication scheme and consistent replicas across an unreliable network is no simple task. To deal with these problems, there has for more than a decade been put considerable research effort into the management of groups of objects/processes and the communication between them. This is referred to as group communication. See [35] for a collection of papers on this issue. Various tool-kits have been developed to facilitate fault-tolerant distributed processing, e.g. Isis and Horus/Electra [5, 36]. Mechanisms for fault-tolerance are also likely to be included in CORBA [31, 38, 39, 41]. Hence, ensuring the dependability of network control, management and service provisioning should be within reach by fault tolerance techniques developed for distributed systems.

So where is the challenge in the smart network context? The fault-tolerance and replication strategies applied must be tailored to the requirements and adapted to the particularities of telecommunications. For instance:

- Network operations and end-user QoS requirements put rather strict real time requirements on many tasks. The replica synchronization is based on protocols ensuring causal order [23], which may be time consuming, especially on non-broadcast media like WANs.

- Many tasks carried out in a network, e.g. reservation of a communication resource, have short processing times compared to synchronization times between replicas, hence an approach must be chosen so an immense overhead can be avoided.
- Only a subset of the tasks performed by the network result in state changes that need to be persistent in case of failure. Some of the Internet's success as a transport network may be attributed to its statelessness. In connection oriented networks, connections in the set-up phase has always been considered expendable when failures occur. For other tasks, persistence of the state in case of failure is of outmost importance, e.g. updates of the home location register in mobile networks.

5 LOGICAL FAULTS¹

The fault tolerance strategies discussed in the previous chapters are aimed at physical faults. Logical faults are however the cause of a substantial part of the failures. Failures due to logical faults may affect every aspect of the service and network operation, from aborting a single service transaction to a complete network outage. The increasing logical complexity and interwovenness of smart networks makes them more prone and vulnerable to logical faults.

In spite of extensive use of fault avoidance and validation techniques, cf. Figure 2, logical faults, e.g. software faults, are embedded into a system during its specification, design, implementation and configuration. These faults stay dormant in the system until a combination of input/use of the system and internal state activates them and causes an error. For a mature system, the expected dormancy period can be several years. In the subsequent operation of the system the error may cause a failure as illustrated in Figure 12. In dependable systems, the various modules/components are sought made robust to errors by audit routines, exception handling etc.

In this presentation, it will not be discussed how logical faults could be avoided and tolerated. The focus is instead put on how logical fault influences the dependability of networks, namely the persistence of errors and error propagation.

1. The term logical fault, and not the more common term software fault, is used deliberately since this type of failures encompass much more than software bugs, e.g. design and specification faults in the overall network operation.

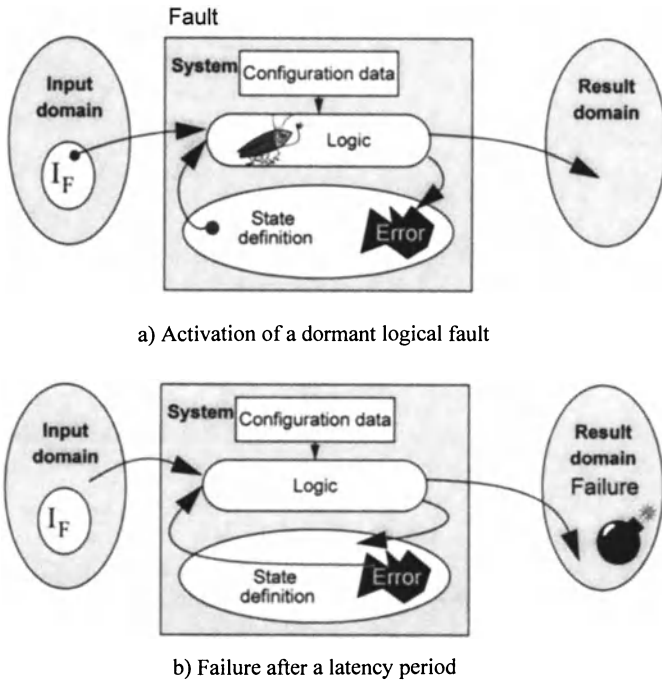


Figure 12. Extended I/O logical failure model based on a generalized Moore/Mealy model.

5.1 Error persistence

A common and rarely disputed assumption in the design and evaluation of dependable systems is that failures occur according to a Poisson process, i.e. failures occur independent of when the previous failures occurred and with a constant occurrence probability per time unit. However, studies of failure logs show that this is not the case, especially for logical caused failures, see for instance [25, 11]. This is demonstrated in Figure 13, which shows the autocorrelation¹, $\rho(j)$, of logically caused failures of a distributed communication control system.

The figure shows the dependency between the number failures in one interval and the number of failures in the j' th subsequent (or previous) interval. (The larger the deviation of $\rho(j)$ is from zero, the greater the dependency.) If the failure process was a Poisson process, the autocorrelation would in 95% of the cases be within the two dotted lines of Figure 13. It is seen that

1. Let $n(t_i)$ be the number of failures in the constant length interval i . The autocorrelation is

$$\text{then defined as } \rho(j) = \frac{\text{Covar}(n(t_i), n(t_{i+j}))}{\text{Var}(n(t_i))} = \frac{E(n(t_i) \cdot n(t_{i+j})) - (E(n(t_i)))^2}{E((n(t_i))^2) - (E(n(t_i)))^2}$$

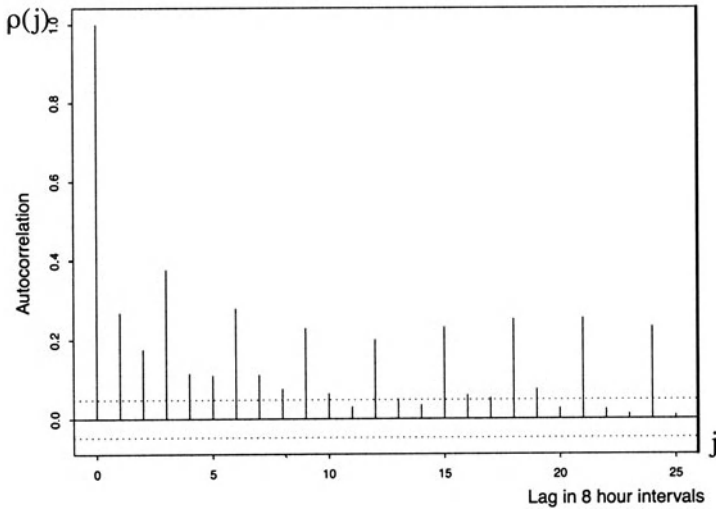


Figure 13. The autocorrelation, $\rho(j)$, between software related failure events in a large distributed communication system.

the failure process has a “memory” of more than a week. Error propagation, as will be discussed in the next section, and changes in operational conditions will also contribute to this correlation. Note also the daily variations seen in Figure 13. The number of failures is more dependent on the number of failures at the same time during the previous day than the number of failures during the previous eight hour interval. Errors may persist in the system for a considerable period and cause a burst of failures, see also Figure 14.

5.2 Error propagation

Figure 13 indicated that errors tend to persist in a system and cause multiple failures. Figure 14 shows a sample from the software related failure pattern of a large distributed communication control system. It is seen that during the first four days we have a normal operational mode with few failures. During the 11th through the 13th day severe bursts of failures occur, affecting several processors. These figures show that the failure pattern is not Poissonian, i.e. failures do not occur independently of each other.

- Failures tend to occur in severe bursts, i.e. if one failure is experienced, the next is likely to occur rather soon. Restarts or similar actions after a failure do not always restore the system to an error free internal state, or the internal state may be inconsistent with the rest of the system or the environment.
- Many processors are involved in a failure burst. If one processor fail, there is an increasing probability of failure of the cooperating processors.

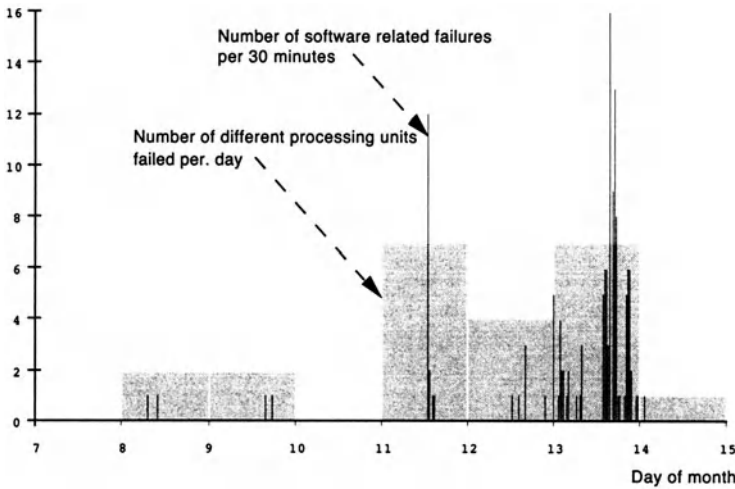


Figure 14. Excerpt of the failure log of a distributed communication control system.

An error located in one part of a software subsystem may propagate (spread) to other parts.

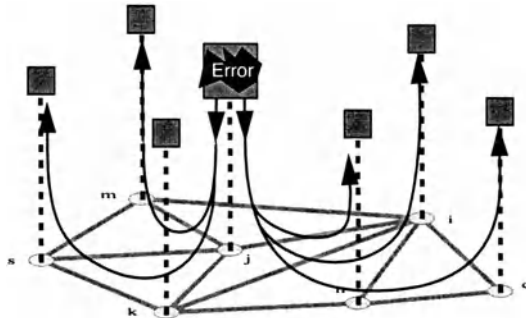


Figure 15. Illustration of error propagation in a network.

For an explanation of the latter phenomenon, regard the sketch of Figure 15. Control/management/service providing subsystems span several physically separate nodes of the network. Each of these subsystems is constituted by a number of more or less tightly coupled software modules residing in the various network nodes. These modules cooperate to carry out the services of the system. Incorrect state information may be transferred from one module to another. For instance by transfer of incorrect or inconsistent data, or by a module acting on false information and thereby introducing errors in its own internal state. Hence, subsequent failures of seemingly independent physical units may occur. This phenomenon is denoted error propagation [12, 13]; see the illustration in Figure 16. The error propagation between modules depends on:

- the intensity of the interactions and
- how much information they exchange/have in common.

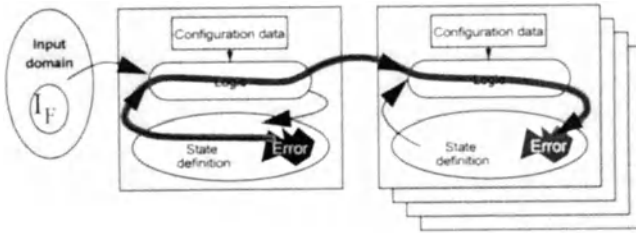


Figure 16. Error propagation a generalized Moore/Mealy model.

A variant of error propagation occurs when the same logical fault is present in identical copies of software modules present in various nodes in the network. These faults may be activated by a condition spreading through the system under specific circumstances, cf. the first well known example [9]. The most prominent logical fault of this kind is the Y2K bug. Analysis of operational data from distributed communication control systems by cluster analysis, shows that the systems may have different operational modes as a result of the error propagation. For instance, “good” modes where only few sporadic failures occur and “bad” modes where many units fail repeatedly within a short time interval [14]. A mode-changing behaviour of the system may be identified. The “bad” modes cripple the service offered severely and may cause instability of the entire network. Furthermore, it is illustrated how the mode-changing behaviour of the system may be described by a state model.

Smart networks are, due to their large logical complexity and interwovenness, exposed to persistent and propagating errors. Hence, to be dependable and have a stable operation, these must have an inherent design which

- enables errors to be removed when a recovery action is taken as a result of a failure;
- prevents error propagation between the various parts of the network.

6 CONCLUDING REMARKS

Dependability is an important aspect of networks in the future. The availability and reliability requirements will continue to increase as our social and economic dependence on teleservices as well as the cost of failure continues to grow. Dependability constitutes a significant part of the QoS and is an important factor in the competition among service providers, network operators and equipment manufacturers. High dependability is costly to provide both with respect to development (fault-tolerance, fault management, and

quality assurance), deployed equipment, and operation and maintenance. At least a doubling of “the basic cost” is required if system shall reach a typical communication network component availability in the order of $1 - 10^{-5}$.

New smart network technologies must be proven as dependable as “traditional” network technologies before they become deployed in commercial networks. Hence, the following basic design question arises; shall proven “dumb” low level (in terms of a layered design) fault handling techniques, like protection switching and duplicated synchronous processors be used, or should new smarter technologies, which are more flexible, and which give potentially cheaper solutions be pursued? Anyhow, the “dumb” techniques are only able to cope with physical faults. The major challenge is the logical (software) faults. The consistent design, design methodology and extensive quality assurance, currently used to restrict the number of logical faults and ensure the dependability, seem feasible only within monolithic proprietary systems as we know them to-day. In the future open, heterogeneous networks with multiple hardware and software vendors, service providers and network operators, robust and adaptive network and service components becomes a necessity. Components able to tolerate logical and physical failures of their environment and co-operating components, adapt and contribute to the provision of continued service is a major challenge in the advancement of smart networks.

REFERENCES

- [1] Finn Arve Aagesen, Bjarne E. Helvik, Vilas Wuwongse, Hein Meling, Rolv Bræk and Ulrik Johansen, “Towards a Plug and Play Architecture for Telecommunications”, This Issue.
- [2] Syed R. Ali, “Analysis of Total Outage Data for Stored Program Control Switching Systems”, IEEE Journ. on Selected Areas in Communications, Vol. SAC-4, No. 7, pp. 1044 - 1046, Oct. 1986.
- [3] Syed R. Ali, “Digital Switching Systems; System Reliability and Analysis” McGraw-Hill, 1997.
- [4] Beyltjens, van Houldt, "System 12, Switching System Maintenance", Electrical Communication, Vol. 59, No. 1/2, pp. 80-88, 1985.
- [5] Kenneth P. Birman, Robbert van Renesse (eds.), “Reliable Distributed computing with the Isis Toolkit”, IEEE Computer Society Press, 1994.
- [6] W.C. Carter, “A time for reflection”, Proc. 12th International Symposium on Fault-Tolerant Computing” Santa Monica California (FTCS-12), p. 41, June 1982.
- [7] Brian A. Coan, Will E. Leland, Mario P. Vecchi, Abel Weinrib, Liang T. Wu, “Using Distributed Topology Update and Preplanned Configurations to Achieve Trunk Network Survivability”, IEEE Transactions on Reliability, Vol. 40, No. 4, pp. 404 - 416, October 1991.
- [8] R.W. Downing, J.S. Novak, L.S. Tuomenoksa, “No. 1 ESS Maintenance Plan”, The Bell System Technical Journal, Vol. 43, No. 5, Part 1, pp. 1961 - 1969, Sept. 1964.

- [9] Karen Fitzgerald, "Vulnerability exposed in AT&T's 9-hour glitch", *The Institute*, Vol. 14, No. 3, Pages 1 and 6, March 1990.
- [10] Jim Gray, "A Census of Tandem System Availability between 1985 and 1990", *IEEE Trans. on Reliability*, Vol. 39, No. 4, pp. 409 - 418, Oct. 1990.
- [11] Bjarne E. Helvik, Anders Rygh Swensen, "Modelling of Clustering Effects In Point Processes. An Application to Failures in SPC-Systems", *Scandinavian Journal of Statistics*, Vol. 14, pp. 57 - 66, 1987.
- [12] Bjarne E. Helvik: "Modelling the Influence of Unreliable Software in Distributed Computer Systems", *Proc. 18th International Symposium on Fault-Tolerant Computing (FTCS-18)*, pp. 136 - 141, June 1988.
- [13] Bjarne E. Helvik, "The Error Propagation Phenomenon; An introduction", *Teletronikk*, Vol. 93, No. 1, pp.109 - 117, 1997
- [14] Bjarne E. Helvik, Sven Arne Gylterud, "Identification of Operational Modes of Distributed Systems by Clustering Analysis", *Teletronikk*, Vol. 93, No. 1, pp.118 - 127, 1997.
- [15] Svein-Olaf Hvasshovd, Øystein Torbjørnsen, Svein Erik Bratsberg, Per Holager, "The ClustRa Telecom Database: High Availability, High Throughput, and Real-Time Response", *Proceedings of 21th International Conference on Very Large Data Bases (VLDB'95)*, pp. 469 - 477, September 11-15, 1995, Zurich, Switzerland.
- [16] ITU-T, "Terms and definitions related to quality of service and network performance including dependability", *Rec. E.800*, August 1994.
- [17] Pankaj Jalote, "Fault Tolerance in Distributed Systems", Prentice Hall, 1994.
- [18] Barry W. Johnson, "Design and Analysis of Fault Tolerant Digital Systems", Addison-Wesley, 1989.
- [19] K.R. Krishnan, R.D. Doverspike, C. D. Pack, "Unified Models of Survivability for Multi-Technology Networks", *Proc. ITC-14* (eds: Labetoulle and Roberts), pp. 655 - 666, Antibes Juan-les-Pins, France, Elsevier, June 1994.
- [20] K.R. Krishnan, R.D. Doverspike, C. D. Pack, "Improved Survivability with Multi-layer Dynamic Routing", *IEEE Communication Magazine*, Vol. 33, No. 7, pp. 62 - 680, July 1995.
- [21] D. Richard Kuhn, "Sources of Failure in the Public Switched Telephone Network", *Computer*, Vol. 30, No. 4, pp. 31 - 36, April 1997.
- [22] Craig Labovitz, Abha Ahuja, Farnam Jahanian, "Experimental Study of Internet Stability and Backbone Failures", *Proc. 29th International Symposium on Fault-Tolerant Computing (FTCS-29)*, pp. 278 - 285, June 1999.
- [23] L. Lamport, "Time, Clocks and the Ordering of Events in Distributed Systems", *Communication of the ACM*, Vol. 21, No. 7, pp. 558-565, July 1978.
- [24] Jean-Claude Laprie (Ed), "Dependability: Basic Concepts and Associated Terminology", *Dependable Computing and Fault Tolerant Systems. Vol-5*; Springer, 1992.
- [25] P.A.W. Lewis, "A Branching Poisson Process Model for the Analysis of Computer Failure Patterns", *Journ. of The Royal Statis. Soc. Ser. B*, Vol. 26, pp. 493 - 503, 1964.
- [26] Michael R. Lyu (ed.), "Handbook of Software Reliability Engineering", McGraw-Hill/IEEE Comp. Soc. Press, 1996.
- [27] Silvano Maffei, Douglas C. Smidt, "Construction of Reliable Distributed Communication Systems with CORBA", *IEEE Communication Magazine*, Vol. 35, No. 2, pp. 56 - - 60, Feb. 1997.

- [28] John J. Metzner, "Reliable Data Communications", Academic Press, 1997.
- [29] L. Nederlof & al. "End-to-end Survivable Broadband Networks" IEEE Communication Magazine, Vol. 33, No. 9, pp. 63 - 70, Sept. 1995.
- [30] Louise E. Moser, P. M. Melliar-Smith, Deborah A. Agarwal, Ravi K. Budhia, Colleen A. Lingley-Papadopoulos, "Totem: A fault-Tolerant Multicast Group Communication System", Communications of the ACM, Vol. 39, No. 4, pp. 54 - 63, April 1996.
- [31] Louise E. Moser, Roger J. Martin, "Fault Tolerance for CORBA" (Joint initial fault tolerance RFP submission by Eternal Systems and Sun Microsystems), <ftp://ftp.omg.org/pub/docs/orbos/98-10-08>, October 19, 1998.
- [32] Bengt Ossfelt, Ingmar Jonsson, "Recovery and Diagnostics in the Central Control of the AXE Switching System", IEEE Trans. on Comp., pp. 482-491, June 1980.
- [33] Vern Paxson, "End-to-End Routing Behaviour in the Internet, IEEE/ACM Transactions on Networking, Vol. 5, No. 5, pp. 601 - 615, Oct. 1997.
- [34] David Powell, "Distributed Fault Tolerance: Lessons from Delta-4", IEEE Micro, February 1994, pp. 36 - 47.
- [35] David Powell (Guest Editor), "Group communication", Communications of the ACM, Vol. 39, No. 4, pp. 50 - 97, April 1996.
- [36] Robbert van Renesse, Kenneth P. Birman, Silvano Maffei, "Horus: A flexible Group Communication System", Communications of the ACM, Vol. 39, No. 4, pp. 76 - 83, April 1996.
- [37] Daniel P. Siewiorek, Robert S. Swarz, "Reliable Computer Systems; Design and Evaluation", Digital Press, 2nd edition, 1992.
- [38] Chris Smith, "Fault Tolerant CORBA" (Fault tolerance joint initial submission by Ericsson, IONA, and Nortel supported by Alcatel), <ftp://ftp.omg.org/pub/docs/orbos/98-10-10>, October 20, 1998.
- [39] Jeffrey R. Spirn, "Fault Tolerance RFP: Initial Submission" (Oracle), <ftp://ftp.omg.org/pub/docs/orbos/98-10-13>, October 20, 1998.
- [40] Paul Veitch, Dave Johnson, "ATM network Resilience", IEEE Network, Vol. 11, No. 5, pp. 26 - 33, Sept./Oct. 1997.
- [41] Shalini Yajnik (ed.), "Fault Tolerant CORBA using Entity Redundancy" (Joint initial fault tolerance RFP submission: Highlander Communications, L.C., Inprise Corporation, Lockheed Martin Corporation, Lucent Technologies, TIBCO Inc., and supported by Academia Sinica, Taiwan), <ftp://ftp.omg.org/pub/docs/orbos/98-10-09>, October 20, 1998.

SERVICE FEATURE INTERACTION

Extended Abstract

Elie Najm

Ecole Nationale Supérieure des Télécommunications - Paris - France

Elie.Najm@ENST.fr

Key words: feature interaction, service, formal methods, development life cycle.

The telecommunications market is witnessing an exponential growth. Deregulation has set the stage for an open and multiple-sided arena of vendors, network operators and service providers. Classical telephony has gained a new momentum through the Intelligent Network initiative. The popularity of internet based communications and of mobile systems has brought new business opportunities. There is, nevertheless, a ransom for the success of telecommunications systems: they are also growing in complexity, the management of which is a challenging issue that must be addressed. Service interaction is one manifestation of this complexity.

1. STATING THE PROBLEM

Simply stated: an interaction occurs when a newly introduced service changes the behavior of existing services. One may distinguish between positive and negative interactions depending on whether the resulting behavior is intended and useful or annoying and harmful. Services are themselves composed of elementary services, called features. Thus, the interaction problem may be revealed both at the service and feature levels.

Feature and service interaction is a complex problem. A large community of experts, from Telcos and academia, is actively working on this issue. A series of workshops is devoted to this topic and many research papers are being published in journals, conference proceedings and on web sites. The following are a few tokens that reflect the difficulty of the problem:

- It is desirable to let users have control on their services: they may customize them and/or integrate them in their applications. This introduces a new dimension to the interaction problem.
- Interaction may occur between services of different users, subscribed to different providers, and deployed on different networks. The analysis phase cannot cover all existing services that are deployed world-wide. Third party mediators may be needed that are responsible for managing and resolving interactions.
- The conception of an interaction-free service is hard because new services added afterwards cannot be anticipated.
- The resolution of interactions between a set of services should not introduce new interactions elsewhere. If checking each combination of 20 features takes 20 seconds then it will take 2 years to resolve the interactions of 20 features.
- Feature interaction is not specific to switch-based architectures. Circular mailing lists (in e-mail) is an obvious example of interactions in internet-based applications. More generally, IP-based telecommunications will bring sophisticated multimedia services. Multimedia services are much more complex than telephony therefore harder will be the interaction problem.

2. DEALING WITH SERVICE AND FEATURE INTERACTION

The issue is dealt with along all the phases of the service development life cycle. First, at the marketing phase, telecom operators need to analyze if the newly introduced services will put their exiting offer in jeopardy (will customers get unsatisfied or confused ?). At conception phase, logical incompatibilities between services are sought and resolved. Detailed analysis phase, where network and computing resources are taken into account, will reveal new interactions, caused by distribution, non atomicity of operations, or resource sharing. Deployment of services should address also the issue: incompatible services should not co-exist on the same network. Subscription of a service by a user is also an occasion where incompatible services should be avoided. Some of the interactions can only be addressed at run-time: Feature Interaction Managers are added to handle the problem on-line.

Formal methods provide means to reason about feature interaction and to help solving the problem. They are endowed with tools that can be used along the development chain. Different formalisms and tools can be used in

different phases, going from the most abstract service models to network aware representations. Each service can be characterized by a set of abstract properties that should be satisfied in the model. Tools can help checking the validity of these properties.

3. CONCLUSION

Feature interaction is a complex problem that may hinder the objectives of standardization initiatives (IN, TINA, Corba, Parlay, ...): the rapid introduction, at reduced cost, of new services. It is a complex software composition problem in a specific context of telecommunications characterized by distribution, heterogeneity and multiplicity and variety of stake holders. Formal methods and associated tools provide some handles to solve the problem. Component-based architectures (Corba 3, EJB, ...) are expected to contribute to the settlement of the issue. A general satisfactory solution is still being actively researched.

REFERENCES

- [1] T.F. Bowen, F.S. Dworack, C.H. Chow, N. Griffeth, G. E. Herman, and Y.J. Lin, "The Feature Interaction Problem in Telecommunications Systems", *Proc. Seventh Int'l Conf. Software Eng. For Telecommunications Switching Systems*, London, pp. 59-62, July 1989.
- [2] E.J. Cameron and H. Velthuijsen. "Feature Interactions in Telecommunications Systems", *IEEE Comm.*, vol. 31, no. 8, pp. 18-23, Aug 1993.
- [3] "Principles of Intelligent Network Architecture", ITU-T Recommendation Q.1201, Geneva, 1992.
- [4] J. Mierop, S. Tax, and R. Janmaat, "Service Interaction in an Object-Oriented Environment," *IEE Comm.*, vol. 31, no. 8, pp. 46-51, Aug. 1993.
- [5] E.J. Cameron, N. Griffeth, Y.J. Lin, M.E. Nilson, W.K. Schnure, and H. Velthuijsen, "A Feature-Interaction Benchmark for IN and Beyond," *IEEE Comm*, vol 31, no. 3, pp. 64-69, Mar 1993.
- [6] K. Kimbler and H. Velthuijsen, "Feature Interaction Benchmark", *Proc. Third Feature Interaction Workshop (FIW'95)*, Oct 1995.
- [7] P. Zave, "Feature Interactions and Formal Specifications in Telecommunications", *Computer*, vol. 26, no. 8, pp. 20-29, Aug. 1993.
- [8] "Feature Interactions in Telecommunications Systems". K. E. Cheng, T. Ohta, Eds. Amsterdam, IOS press, Oct 1995.
- [9] P. Combes, M. Michel, B. Renard, "Formal Verification of Telecommunication Service Interactions using SDL methods and Tools", *Proc. Sixth SDL forum, 1993 (SDL'93)*, O. Faergemand and A. Sarma, Eds., pp. 441-452, Elsevier Science, 1993.
- [10] "Feature Interactions in Telecommunications Systems". L. Bouma, H. Velthuijsen, Eds. Amsterdam, IOS press, May 1994.

- [11] B. Stepien, L. Logrippo, "Feature Interaction Detection Using Backward Reasoning in LOTOS", Protocol Specification Testing and Verification XIV, S.T. Vuong, S. T. Chanson, Eds., pp. 71-86, London, Chapman & Hall, 1995.
- [12] K. J. Turner, "An architectural foundation for relating features", Feature Interactions in Telecommunications Networks, vol. 4, P. Dini, R. Boutaba, L. Logrippo, Eds., pp. 226-241, Amsterdam, IOS press, June 1997.
- [13] O. C. Dahl, E. Najm, "Specification & Detection of IN service interference using LOTOS", Formal Description Techniques, vol. 6 (FORTE'93), R. L. Tenney, P. D. Amer and M. U. Uyar, Eds., pp. 53-69, IFIP trans., North Holland, Oct 1993.
- [14] D. O. Keck, P. J. Kuehn, "The feature and service interaction problem in telecommunications systems: a survey", IEEE trans. On Soft. Eng., vol. 24, no 10, pp. 779-796 Oct 1998.

DEVELOPING A DISTRIBUTED MEETING SERVICE TO SUPPORT MOBILE MEETING PARTICIPANTS

Seamus Hickey MEng

Department of Information Processing Science, University of Oulu, Finland

Key words: Distributed meetings, Mobile Services, Telepresence

Abstract: Future 3rd+ generation mobile wireless systems are been designed to support high data rates in comparison to existing voice based wireless systems. The success of these new systems will partly be based on the data services it can provide. Apart from supporting traditional data (email, ftp etc.) & multimedia applications (videoconferencing, web access etc.), the combination of *mobility* and high speed data terminals will lead to innovative new data services e.g. navigation. In the PAULA project we are implementing an experimental distributed meeting service, based on Telepresence. The service attempts to compensate for the 'in transit' state of a mobile user by enabling them to attend a remote meeting while providing a high level of immersion, or sense of presence, at the meeting. The experimental system comprises of an omnidirectional camera placed in the meeting room that provides a constant navigable video stream to a remote mobile participant utilising a Head Mounted Display.

1. INTRODUCTION

Current mobile phone systems have predominantly focused on voice services while providing only basic data rates up to 14.4 kbps. Considerable interest has now focused on providing higher mobile data rates to support basic data services (e.g. email, ftp) and multimedia applications. Towards this end the introduction of the GPRS data service in 1999 will provide data rates up to 170 kbps [1]. In addition, research and standardisation of 3rd

generation mobile services (e.g. UMTS) is likely to facilitate data rates up to 2 Mbps for indoor systems and 384kbps for outdoor [2] by 2002. With the introduction of these new data rates, the mobile user should not only have access to most of the services available from a fixed computer or workstation, but also have these services anytime and anywhere. In essence, the mobile user has the additional service of *mobility*. The combination of high data rates and mobility can lead to innovative new services like navigation [3]. These new services can also provide support to a mobile user who would normally be disadvantaged while been in transit between destinations or physically located outside of the normal working environment. The service described in this paper relates to supporting such a user, who must attend a meeting at a remote venue and to develop the service sufficiently to compensate for the spatial disadvantage. In this paper we concentrate on the nature of this type of service and will use the knowledge obtained to determine the supporting network requirements.

Distributed meetings, both formal and informal, have attracted considerable research interest. Various different methods have been proposed with a wide range of technical sophistication [4]. The various systems include simple teleconference systems to full Collaborative Virtual Environments (CVE). Until recently, the point of access of these systems has been fixed, e.g. workstations or desktop computers, and very little attention has focused on support for mobile participant's [5]. In general the distributed meeting models under research consists of two types, those requiring all the participants to be distributed and convene in a virtual setting, or specialised meeting room environments with participants communicating over dedicated links. These models do not cater for a more normal situation where the majority of the meeting participants exist in a normal meeting room setting, but a handful of desired participants are remotely located. Traditionally the only manner in which these remote participants could attend was by phone. However, even with normal video conferencing, such methods are not entirely satisfactory as the remote participant is deprived of many normal interactions available by been physically present, preventing them from been as effective as if they were they present [6,7].

To compensate and provide a higher sense of immersion to the mobile user, a distributed meeting service based on telepresence techniques is proposed. An experimental distributed environment has been implemented which helps to give a higher sense of immersion than is normally available from video conferencing systems. This paper describes the components of the system, provides other research that supports this work, describes the current experimental system, discusses the results and identifies future work and improvements to refine the service.

2. CURRENT DISTRIBUTED MEETING MODELS

Current research models for distributed meetings have focused on 4 areas, media spaces [8,9,10], spatial video conferencing (11,12.), Collaborative Virtual Environments (CVE) [13,14,15], and telepresence [16,4].

Media Spaces. Media Spaces enhance an existing workplace (e.g. an office) by using audio/video services as a basis for providing a range of general services. These services support social browsing, peripheral relationships and the establishment of long term relationships between physically separated people. Examples of such systems are RAVE [10] and Cruiser [7]. Experiences with the Cruiser system indicate that this medium was used primarily as a precursor to longer meetings in which all the participants were co present.

Spatial Video Conferencing. Spatial Video Conferencing covers video and audio communication over a desktop system or through dedicated links between public meeting rooms. These Video conferencing systems are sometimes enhanced with support for collaborative and shared work spaces. Video Conferencing systems between dedicated meeting rooms involve a significant support and logistics burden, which creates an obstacle to its adoption [12]. Furthermore, the lack of gaze direction, an important element in meeting environments, is a major disadvantage to such systems. Solutions to the gaze awareness problem have led to systems like MAJIC [11]. However MAJIC is limited to a maximum of 4 meeting participants.

Collaborative Virtual Environments (CVE). CVEs are networked virtual reality systems where each meeting participant is represented by an avatar, a virtual graphical embodiment of a real person, in a virtual reality meeting room. Within the virtual meeting room, the participants may engage in social and collaborative work. Experience with CVE's [14] has highlighted that extensive collaborative work is needed to first set up the meeting. Also, the embodiment does not accurately reflect the state of its 'real' meeting participant, e.g. the participant may have left the office or is performing another task, yet the embodiment still exists in the virtual meeting room. This can lead to confusion between the meeting participants.

Telepresence. Telepresence allows users to experience a remote physical space through computer and communication technologies. Typically, the person is represented in the remote space by a physical proxy e.g. robot. The user may experience a good level of immersion by using a Head Mounted Display (HMD), where live video is displayed instead of 3d graphics. One of the most notable examples of such a system is the GestureCam [16].

3. TELEPRESENCE IN DISTRIBUTED MEETINGS

From a review of the current research into distributed meetings, it would appear that the results from the use of a Telepresence system could provide the best means of communication support for a mobile user. In the design of such a service, it is important to highlight the major factors that should be considered in the design.

The GestureCam[16] experiment comprised of a remote user attending a meeting whereby he is attempting to instruct the meeting participants on how to operate a piece of equipment. A camera was mounted on an actuator and placed in the meeting room. The camera was controlled by the remote user, who instructed it to move in which ever direction he so wished. Some major conclusions from this work indicated that an additional camera was required to provide a wider angle of view. A similar problem was encountered with Media spaces [17]. In the MTV experiment, it was noted that there were '*significant constraints due to the limited field of view offered by a single fixed camera*' and it concluded that '*techniques that allow continuous camera movement in remote spaces might overcome the limitations we observed in this study*'. These conclusions argued well for the use of physical proxy.

There are a number of problems with using a physical proxy. Apart from the obvious problem of maintaining a mechanical device, in practice, only one person can have control of the proxy at any one time. Controlling who would have control of the view is problematic enough, yet it is certain that the view that is suitable for one person may not suit another.

An important factor to consider in the design of a telepresence system is the significant problems that can occur with the meeting setup. In general, the complexity or difficulty in establishing the meeting should not be such, that it would discourage people from using it.

In an attempt to counteract these problems an omni-directional camera, which has a Field of View (FOV) of 360 degrees, is used as a physical proxy. An omni-directional camera can provide a 360 degree FOV, while been fixed in one position. This removes the need for maintaining a mechanical device. In addition, with the use of supporting software the camera can support multi-user access.

3.1 Distributed Meeting Service Architecture

The distributed meeting architecture is shown in Figure 1 and is comprised of a number of components. The premise of the architecture is to support a mobile user who wishes to attend a meeting at a remote location. The components of the architecture are based on a mobile user, who is

utilising an advanced mobile terminal with support for a HMD, service profiles sitting on top of a bearer service (e.g. WLAN), and a meeting room which contains a camera capturing device and bi-directional audio conferencing equipment. The mobile user and the meeting room terminal each has it's own service profile, with the meeting room terminals profile been controlled by the provider of the meeting room.

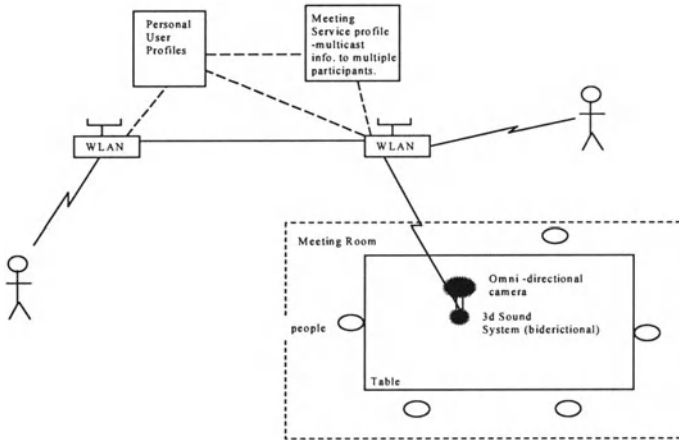


Figure 1. A Distributed meeting service architecture supporting multiple users

3.2 Users Mobile Terminal

The mobile data terminal required to support the distributed meeting service described so far, requires an advanced data terminal capable not only of providing high data rate access, but has a sophisticated display technology, such as support for a HMD. While no such terminal currently exists commercially, considerable research effort is underway into realising this technology. The CyPhone [18] is one of the first instantiations of such a terminal and is shown in figure 2(a). A future iteration of the terminal is shown in figure 2(b). It is conceivable that such terminals can support Telepresence and Augmented Reality services. It is also anticipated that in the next decade or so, that HMD's may be as widely available as Walkman's are today [19]. The popularity of such devices may result from people preferring to use these devices instead of a more cumbersome laptop computer and indeed observations of people at meetings showed them using Personal Digital Assistants (PDA) in preference to laptops [20].

3.3 Service architecture

In line with the current trend in mobile communications, the mobile user will have their own personnel service profile. The service profile will have features that indicate its ability to use advanced visual services like the distributed meeting service. In our current setup, the user service profile is trivial. The meeting room terminal's service profile may contain considerably more service features. Not only would it control user access to the meeting, but it could provide additional services such as recording minute meetings, storing data, scheduling etc. The bearer service used is a 2 Mbps WLAN. This was chosen to mirror the anticipated bandwidth that will be available from UMTS.



(a)



(b)

Figure 2(a). An example of a CyPhone with stereoscopic display.
2(b). A proposed future version of the CyPhone with support for a Head Mounted Display.

3.4 Meeting room terminal

The data terminal that occupies the meeting room contains an image capturing devices and bi-directional audio conference equipment. The preferred image capturing device is an omni-directional camera that is capable of taking a 360 degree video stream of the surrounding environment.

4. DISTRIBUTED MEETING TEST ENVIRONMENT.

An experimental distributed meeting service has been established at the University of Oulu. The experimental system is based on the model shown in figure 1, with a number of differences. Firstly, the project does not use a

sophisticated data terminal like the CyPhone. Also, the service structure has not been implemented entirely, only the WLAN part has been used in the experiment.

The meeting room setup consists of the CycloVision ParaCamera, which is connected to a Matrox Meteor II frame grabber that continuously grabs a constant stream of video images. The CycloVision ParaCamera shown in figure 3, is comprised of the Panasonic colour digital camera GP-KR222 and an optical system comprising of special lens and a parabolic mirror that captures a 360-degree view of the surrounding area. The Field Of View (FOV) of the ParaCamera is 360 degrees horizontally and 180 degrees vertically. The captured image is called an 'obstructed image', as the image is partially obstructed by the camera itself and the metal holding the mirror. Each recorded frame is called a ParaFrame and shows the entire 360 degree view in one frame. This is difficult to interpret by a normal human user. Additional software is required to convert the ParaFrame into a normal 'perspective' view. This perspective view provides a narrower FOV and the video image is that which would be expected from a normal video camera. Additionally, it is possible to dynamically change the perspective using a mouse or head tracking system to navigate the ParaFrame. CycloVision provides ParaPlayer software to convert the 360-degree video stream into a perspective frame. However, this software was not suitable for our purposes because it did not support a full screen mode, which is essential to provide a sense of immersion using the Head Mounted Display. Therefore new proprietary software was developed to perform the perspective transforms in full screen mode. One side affect of converting the images to a full screen mode, is that it has a degrading affect on the image resolution.

The ParaCamera was placed on the corner of the table. Originally it had been intended to invert the camera and suspend it from the ceiling, i.e. with the parabolic lens towards the ceiling. However, finding the optimal height to suspend the camera was problematic. This optimal position for viewing participants sitting down had the side affect of 'cutting off' the upper body of a standing participant. Manually changing the camera position to compensate was not considered as an option because it would require excessive fidgeting which would prove an annoying diversion to achieving the meetings objective. Consequently the camera was placed with the parabolic lens lying on the table, as shown in figure 4. This solution was not entirely satisfactory, as the video image suffered due to interference from light sources e.g. light bulbs, sunlight. The camera can be modified to compensate for excessive light intensity.

The ParaCamera shown in figure 4 is the spatial representation of the remote mobile user in the meeting room. Initially it had been intended to place it in the centre of the room, but to provide a more natural feeling, it

now approximates the position a person would take were they to physically attend the meeting

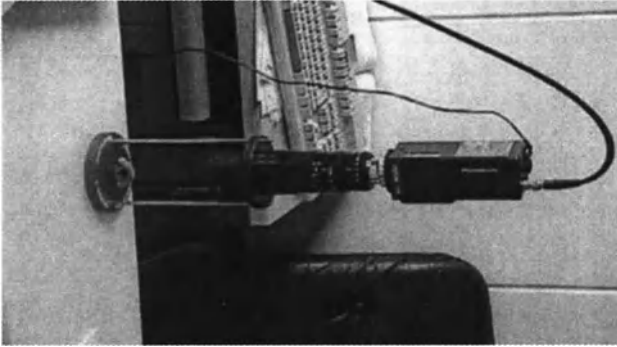


Figure 3. A CycloVision ParaCamera, consisting of a colour camera, specialised optics and a parabolic mirror.



Figure 4. The meeting room with two participants. The ParaCamera is placed on some books

The mobile user was located in a second room on the same floor, but approximately 20 meters away. The mobile user wore a Head Mounted Display. The Head Mounted Display, shown in figure 5, was attached to a computer and connected to a Polhemus tracking system. The Polhemus tracking system is required to monitor the exact head position of the remote user. The head position co-ordinates are transmitted to the computer where they are used to calculate the perspective view for the mobile user. The perspective view is then displayed on the HMD. By moving his head the remote user can browse the remote meeting room.



Figure 5. The mobile participant with a Head Mounted Display.

The meeting room video stream is transmitted to the remote user's computer over a WLAN. It is important to mention that the omni-directional video stream is transmitted over the WLAN and at the remote user's end, the perspective image is calculated based on their head.

The environment was tested during one project meeting. The meeting consisted of six people, with one person as the remote participant. While initially there was considerable interest in the technology, the meeting managed to progress normally, despite a number of intrusions by curious onlookers. This affected the remote user more as people were unaware that he was attending a meeting.

5. DISCUSSION

The initial feedback from the trials has been encouraging. The ability of the remote user to look in whichever direction they desired was a strong point of the system. Users commented that they felt a higher sense of immersion than they had expected. The added advantage of getting visual feedback, or cues, from the other meeting participants in the remote meeting, was considered to be an improvement over audio only based communication, results are in line with other research work [7].

Observations of the remote user showed them changing their head positions in order to look around the room or focus on an individual in the remote meeting. The use of a HMD with live video feed was considered to be more 'immersive' than viewing the same video stream on a computer monitor.

The system as it presently exists has a number of both technical and perceptual problems, partially caused by the software, the camera system chosen and some general weaknesses in the system.

Implementing our own perspective transforms to provide a full screen mode for the HMD was achieved at the expense of resolution. The resolution

was sufficiently poor so that the remote user had difficulties in recognising people beyond a distance of approximately 2 meters. While this was adequate for our meeting setup, which was based on a small meeting group, it is not adequate for larger meetings, or capable of supporting presentations. In our case, not many users seemed to be unduly concerned about the picture quality, as it was purely a discussion based meeting.

The main user comments centred on the frame rate, which was very slow, especially when users changed their head position quickly. It usually took a few seconds for the correct perspective view to correspond with the users head position. The video images were transmitted without compression and while we have developed compression algorithms, these were not working with the software at the time of these tests. However, the integration of the compression algorithms should improve the frame rate.

The affect of light sources on the camera considerably degraded the image at times. The mobile users complained about turning to see a person to their right, who was sitting by a window, and of being unable to see them properly because of the sunlight through the window. Consequently, it was necessary to draw the blinds in the room to improve the picture quality. While the camera was set for the average light intensity, there seemed to be 'light spots' which had a higher intensity of light in comparison to the normal light intensity.

Apart from the technical problems, a number of communication and perceptual problems occurred. One user complained that the original position of the camera, lying on the table, made him feel as though he was 'looking up' at everybody else. He asked us to raise the camera to his perception of eye level by placing some books under the camera. Additionally, a problem occurred when the mobile user was attempting to decide who was currently speaking. In a discussion afterwards, the mobile users commented that it was easier to know who was talking depending on their ability to recognise different voices. Those unable to recognise a meeting participant's voice had a harder time discerning that speaker. It was noted that the slow frame rate and poor resolution did not help pick out who was talking, but observing the direction in which the other meeting participants were looking did help.

From the viewpoint of those physically attending the meeting, it was noticed that the introduction of the system had little affect on their normal behaviour. These participants were largely acting as though it was a normal phone conference, as they only had an audio connection. These meeting participants had a number of observed problems with the audio system. On a number of cases they were forced to ask if the remote participant was still there as there was no feedback from him. On one occasion, the mobile user was showing the HMD to another person who had just walked in to the

room, which was unknown to the people in the meeting room who were trying to get his attention. This is a general problem with audio only communication and also recorded by those working with CVE's.

On another occasion, the mobile user attempted to verbally communicate information (an email address) to the meeting room participants. To clarify the information, the meeting participants wrote out the given email address on a large sheet of paper that they showed to the camera in order to confirm that it was correct. It can be concluded that the efficient transfer or sharing of information is a considerable weakness in the system that needs to be addressed.

While the meeting participants generally treated the remote mobile user as a normal phone user, it was observed that they often turned to the camera to address the user. Of course this phenomena could also be the result of addressing the audio connection which was co-located with the video camera. The affect of doing this was quite natural to the users who seemed to relate the camera to the 'presence' of the remote mobile user.

6. IMPROVEMENTS AND FUTURE WORK.

The experimental distributed meeting environment described in this paper has given us initial feedback and results which highlight both the advantages and problems in this service concept. The feasibility of the service depends on 2 criteria, the anticipated wide availability of cheap HMD, and adding sufficient features to the service so as to encourage mobile users to select it as the preferred method of communication [21]. With this in mind, we are moving to firstly improve on the system by taking the experimental results into consideration and solving the immediate problems.

The distributed meeting environment described in this paper requires some modifications to improve its performance and usability. The most immediate work will concentrate on implementing the compression algorithms to improve the frame rate. This was the primary issue raised by those who tried the system.

The main computing complexity resides at the remote user's location, as this is where the perspective transforms are calculated. In real terms this would mean that the mobile terminal would require significant computing power, increasing the expense of the terminal. Alternatively, the perspective image could be calculated in the network backbone. In this scenario, the mobile terminal will send the head position co-ordinates to the backbone which computes the perspective image. A comparison of the two techniques in terms of benefits and efficiency would be interesting.

In addition, the current environment only supports a single user. Expanding the system to support multiple mobile users should give us some idea of the problems they face. The implementation of a protocol to manage multiple user support is under development.

To solve the problem of the poor video resolution will require a more powerful omni-directional system. An 'Image Based Rendering' system using an omni-directional camera comprising of a matrix of 8 cameras and morphing software has been developed at the University of Tokyo [22]. This system offers a significant improvement on resolution but does not perform in real-time as yet. However, the approach taken provides a possible solution to the poor resolution offered by the ParaCamera.

As the audio connection is also of major importance, a 3D sound system will be added to assist in the location of a speakers voice and to provide a deeper sense of immersion. To provide some form of visual user feedback to the meeting room participants, we will explore the use of 'talking heads' displayed on a monitor at the location of the omni-directional camera. Another solution could involve having a miniature camera as part of the mobile user's data terminal, as suggested in the CyPhone project, providing a video stream to the meeting room participants.

Improving collaborative work and the transfer of data will require innovative new techniques. Most of the existing collaborative techniques are not suitable for mobile users unless they have a laptop, or some access to a desktop computer. One innovative new method that will be explored, as a possible solution, is to incorporate 'augmented reality' objects in the meeting room [23]. Augmented reality objects are computer graphical objects that can be superimposed upon the video stream. A user can interact with these objects to perform functions or tasks. These objects can represent various items in the meeting room, which can be selected by the mobile user and enable him/her to send and receive information e.g. slides, emails etc.

7. CONCLUSIONS

A distributed meeting service supporting mobile users has been introduced. A possible service architecture has been described based on Telepresence and the justification for this work and approach taken has been provided. The advent of high mobile data rates and sophisticated data terminals supporting Telepresence and Augmented Reality techniques supports the feasibility of this service.

An experimental telepresence system using a CycloVision omni-directional ParaCamera and a remote user connected to a HMD have been demonstrated. The system has been shown to provide a more flexible ability

to view the surrounding area of a remote meeting and is capable of supporting multi-user access. Results have been encouraging and the plausibility of the system has been sufficiently justified to merit further research and expenditure of resources.

Furthermore, areas of improvement have been identified. Of priority will be improving the frame-rate by adding compression algorithms, an efficient broadcast protocol to provide multi-user access, a 3D sound system to provide a deeper sense of immersion, support for 3D talking heads and incorporating augmented reality objects to enable user interaction with the environment.

7.1 Acknowledgements

This project is sponsored by the PAULA project. The PAULA project is supported by the Academy of Finland and is a co-operative work between the University of Oulu, Helsinki University of Technology, VTT Oulu and Tampere University of Technology. I would also like to thank Tino Pyssysalo, Prof. Petri Pulli, Sami Ahola and Prof. Kari Kuutti for their invaluable assistance during this project.

REFERENCES

1. Concepts, Services and Protocols of the New GSM Phase 2+ General packet radio Service. G. Brasche, B. Walke. IEEE Communications magazine, August 1997, Vol 35. No. 8. IEEE Communications Society
2. Research Activities on UMTS Radio Interface, Network Architecture and Planning. E. Berruto, M. Gudmundson, R. Menolascino, W. Mohr, (1998). IEEE Communications Magazine, Feb. 1998, Vol. 36 No. 2
3. Services and Applications: Requirements and realisations in the UMTS Era. A. Munro, E. Geulen, M. Deguine, G. Melpignano. 1998 IEEE Communications Magazine, Feb. 1998, Vol. 36 No. 2
4. Shared Spaces: Transportation, Artificiality and spatiality. S. Benford, C. Brown, G. Reynard, C. Greenhalgh. Proc. Of ACM 1996 Conference on CSCW. Boston, Massachusetts.
5. Mobility in Collaboration. P. Luff, C. Heath. Proceedings of ACM 1998 conference on CSCW. ACM Press.
6. A comparison of face to face and distributed presentations. E. Isaacs, T. Morris, T. Rodriguez. Human Factors in Computing Systems, CHI '95 Conference proceedings. ACM Press.
7. Evaluating video as a technology for Informal Communication. R. Fish, R. Kraut, R. Root. Human Factors in Computing Systems, CHI '92 Conference proceedings. ACM Press.
8. A Virtual Window on Media Space. W. Graver, G. Smets, K. Overbeeke. Human Factors in Computing Systems, CHI '95 Conference proceedings

9. Portholes: supporting awareness in a distributed Work group. P. Douish, S. Bly. Human Factors in Computing Systems, CHI '92 Conference proceedings. ACM Press.
10. Realising a video environment: Europarcs RAVE system. W. Gaver, T. Moran, A. MacLean. Human Factors in Computing Systems, CHI '92 Conference proceedings. ACM Press.
11. Multiparty Videoconferencing at Virtual social distance: MAJIC design. K. Okada, F. Maeda, Y. Ichikawaa, Y. Matsushita. ACM 1994 Conference on CSCW, Chapel Hill, North Carolina. ACM Press.
12. Automated Assistance for the telemeeting Lifecycle. N. Bergmann, J. Craig Mudge. ACM 1994 Conference on CSCW, Chapel Hill, North Carolina. ACM Press.
13. A Distributed Virtual Environment for concurrent Engineering. J. Maxfield, T. Fernando, P. Dew. Virtual Reality - Annual International Symposium 95, IEEE Computer Society Press.
14. Practically accomplishing Immersion: Cooperation in and for Virtual Environments. J. Bowers, J. O'Brien, J. Pycock. Proc. Of ACM 1996 Conference on CSCW, Boston, Massachusetts, ACM Press.
15. MASSIVE: A Collaborative Virtual Environment for Teleconferencing. C. Greenhalgh, S. Benford. ACM Transactions on Computer-Human Interaction, Vol. 2, No. 3, Sept. 1995.
16. GestureCam: A video communication system for sympathetic remote collaboration. H. Kuzuoka, T. Kosuge, M. Tanaka. ACM 1994 Conference on CSCW. ACM Press.
17. One is not enough: Multiple Views in a media space. W. Garver, A. Sellen, C. Heath, P. Luff. Human Factors in Computing Systems, INTERCHI '93 Conference proceedings. ACM Press.
18. CyPhone - Future Personal Telecooperation Device. P. Pulli, T. Pyssysalo, J. Similä, J. Metsävainio. Telecooperation. Proceedings of the XV. IFIP World Computer Congress, 31/8 - 4/9 1998, Vienna & Budapest.
19. An Application of Augmented Reality in Future Mobile Communication services. T. Pyssysalo. Personal Wireless Communications.
20. Collaboration using multiple PDAs connected to a PC. B.A: Myers, H. Stiel, R. Gargiulo. Proceedings of ACM 1998 conference on CSCW- edited by S. Poltrock and J. Grundin. ACM Press
21. Beyond being There. J. Hollan, S. Stornetta, Human Factors in Computing Systems, CHI '92 Conference proceedings. ACM Press.
22. Building a Virtual World from a Real world. M. Hirose, T. Tanikawa, T. Endo. Mixed Reality. Merging Real and Virtual Worlds, Edited by Y. Ohta, H. Tamura.
23. A taxonomy of Real and Virtual World Display Integration. P. Milgram, H. Colquhoun. Mixed Reality. Merging Real and Virtual Worlds, Edited by Y. Ohta, H. Tamura.

VOICE SERVICES FOR PSTN AND IP NETWORKS

Qi Guan

SIEMENS AG Austria

Siemensstraße 88-92, A-1210

Vienna, Austria

Key words: Voice over IP, VoIP, Services, PSTN

Abstract: This paper presents an architecture for Voice over IP services between PSTN and ISPs. The architecture is vendor independent or its vendor dependence is minimal. That means it is easily applicable for ISP's and LEC's. With this system the called party is called with his original PSTN phone number. The call is also roamable so that the called party can be reached independent of his location, of VoIP terminal or even of his ISP's. The administration of these services is configurable either by the traditional way or on the web. Thus, all these can increase the call completion rate. This benefits the IP users from not missing any important telephony calls during an Internet session. It benefits the PSTN carriers to generate revenues from both telephony calls and Internet on-line access usage at the same time. It benefits the ISPs from providing the value-added service to increase their profitability.

1. INTRODUCTION

On-line access becomes more and more popular in everyday life. Residential Internet subscribers access mostly the Internet by dialing into the Public Switched Telephone Network (PSTN). A study, which published in 1997 [1], indicates that the Internet users have different behavior than usual PSTN users. The traffic generated by these Internet users is also vastly different from the traffic traditionally observed on the PSTN. It results that the local exchange carriers (LEC) are facing new challenges in adapting their networks. The Internet Service Providers (ISP), which satisfy the growing

mass market of on-line access, are facing the great challenges to become profitable since the number of ISPs continue to proliferate.

To grantee successful Internet access, ISPs must provide enough lines. While the variation of number of Internet access is large, peak load of ISPs usually only happens in the evening. This means most of the time the line side equipment (modem banks) of ISP's are not utilized efficiently.

The Internet access by dial-up lines has significant impacts on the PSTN traffic. It increases the number of busy signals per line. Even during day-time, the traditional voice busy-hours, the increase Internet traffic is considerable. This requires the LECs to rearrange of subscriber-line in concentration-stages, to increase the capacity of trunk equipment, even to purchase additional ones. On the other hand, busy terminating lines reduce the revenue of the operator, since he has to provide expensive call setup resources that do not end in completed calls.

Both local exchange carriers (LEC) and Internet Service Providers (ISP) have recognized that it is necessary to offer value-added services. The Internet telephony or more precisely Voice over IP (VoIP) is one of the new exciting opportunities for both LECs and ISPs. This new service helps LECs to hold its existing end-users, to attract new customers, to increase the telephone call completion rate, to improved the efficiency of the PSTN. For ISPs this will improve their marketing position and profitability by serving these features.

In this paper, we present a new VoIP architecture to integrate Voice over IP services between LEC of PSTN and ISPs. With this architecture more calls can be completed even while the user is in an Internet session. Call completion is an important aspect of the service provided by telecommunication operators. Calls terminate with busy or no-answer state, consuming network resources. Resolution with combination of call forwarding and Internet Call Completion contributes to greater call completion which lowers operating expense and provides better value to both the consumer and service provider. This architecture is for LEC minimal vendor dependent. It provides PSTN supplementary services like call forwarding etc., for both PSTN user and IP user.

2. PSTN/INTERNET INTERWORKING

The PSTN/Internet interworking is one of the most rapidly developing fields in the telecommunication industry. It allows Internet user (or IP user) to dial up from PSTN terminal to the Internet world. With the Voice over IP (VoIP), it becomes possible to make calls from/to PSTN user while an Internet session is being engaged.

At present, a well known standard suite for VoIP is ITU H.323 protocol [2]. In H.323 four main components (called entities) are defined: Terminals, Gateways, Multipoints Control Units and Gatekeepers. An H.323 Gateway (GW) is the voice or multimedia internetworking component for connecting IP packet-based network and PSTN. An H.323 Gatekeeper (GK) is located on the Internet that provides address translation and controls access to data network for H.323 Terminals, Gateways etc. The gatekeeper may also provide other services to the Terminals, gateways and MCUs such as bandwidth management and locating Gateways.

There is another protocol to handle VoIP, this is the Session Initiation Protocol (SIP)[3] defined by IETF.

A IP user in an Internet session that uses a VoIP terminal can dial an E.164 (a usual telephone number system) code to call a PSTN user. This number in most cases must be dialed including the international code and area code. The international code or area code is needed because that the GK or SIP Server is like to build a VoIP call connection to the nearest gateway from called PSTN user.

There are many discussions for a PSTN user to call a IP user.

ETSI-TIPHON and ITU are currently investigating to assign an International-Service-Code (a.k.a. Country-Code) for IP-Telephony. This new E.164 code than could be dialed by any PSTN phone to reach a IP terminal, or dialing a gateway and the IP-address of the terminal in decimal 4*3 digit notation. Dialing the gateway and a randomly assigned number is also required by some implementations.

Yet another way is that a PSTN user dial the number of the GW/GK as prefix plus the calling code of Internet user which connected to the telephone line. To become reachable, the IP user will have to first register himself to certain servers, like Gatekeeper (GK) and Location Server (LS), to indicate that he is in an Internet session and the VoIP program is running in his

computer (possible in a background mode). When a PSTN user dials an IP user, the GK will look up the LS for the status of the IP user. If the IP user is currently engaging an Internet session, the GK will establish the connection to PC and GW sends the converted voice packets to computer. The VoIP program is activated.

The drawbacks of the methods mentioned above are obviously for both caller and called party:

- The caller doesn't know whether the called party is in the Internet session. A caller may at first dial directly to the called party per PSTN. If it is busy, the caller may dial to GW/GK. If called party is in Internet session and has been registered in GK, the call connection can be established. If the called party is not in Internet session, the call to GK/GW will also fail. Thus increases the busy traffic.
- The called party must make known his original telephone calling code and the calling code of GW/GK.
- The calling code of GW/GK usually depends on ISP (Internet Server Provider). If the called party has changed his ISP, the related GW/GK code must also be changed.

3. A NEW VOIP ARCHITECTURE

In this section we present a new architecture of VoIP services for interworking between PSTN LEC's and ISPs. The advantages of the system are

- The called party is being reached with his original PSTN calling code (a.k.a. phone number). The working state of called party, the ISP, and the GW/GK is transparent for caller. Neither the called status, nor the GW/GK number is a concern of caller.
- The call is roamable, which means the called party can be called, under a single, location-independent address, while the user may use different VoIP terminals on different machines in different locations, even with different ISPs.
- The interworking implementation between PSTN and ISP is PSTN vendor independent.

- The administration of services on PSTN side is configurable. It provides a means by either the traditional way or the web-based application.
- The services are available for either caller or called party, either PSTN user or IP-users. The caller is not aware of whether the called party is in a PSTN or in an Internet session.
- It provides usual PSTN supplementary services like call-forwarding (CFD) for both PSTN and IP-users. Other PSTN supplementary services like absent subscriber services may also be provided for IP-users.

3.1 Overview

Figure 1 shows the overview of the architecture of integrated VoIP services for PSTN and ISPs. The components are:

- Management Service (MS) and RADIUS server;
- Presence Service (PS) and Forwarding Service (FS);
- VoIP gateway;
- Location-server/H.323 Gatekeeper (GK) or the Session Initiation Protocol (SIP) server;
- Internet Subscriber Controlled Input via Internet (ISCI) Service;
- Switched Communication Network (SCN). As option, PSTN switches in SCN may have connection to Intelligent Network (IN);
- ISPs.

The Management Service (MS) is responsible for Internet user administration such as configuration and management, billing, user access right. In the Management Service a RADIUS client may be started in order to exchange information with RADIUS server.

RADIUS (Remote Authentication Dial In User Service) server is IETF RFC 2138 and RFC 2139 compatible. It provides functions as security, authorization and accounting. It allows for authentication (verification of user name and password) as well as configuration information (user profile), describing the type of service subscribed by the user.

This registration is done when a user logs in to the Internet. The Management Service should verify whether the user is allowed to access this VoIP services. Therefore, the Management Service calls RADIUS to check the authentication of the user. If the user has the right to use the VoIP services, RADIUS responds a positive acknowledge. The user then makes a registration to the Gatekeeper. After the registration, the Management Service will make an IP-address available for the Internet user and store this IP-address in Presence Service(PS). The PS propagates the IP-address to Gatekeeper or SIP server and Forwarding Service (FS) if necessary.

■

The Internet user then can open a web browser to connect to the Subscriber Controlled Input via Internet (ISCI) Service. After authentication, the user can change his call-forwarding attributers in the subscriber database. For example, he can set his call-forwarding on busy or no reply if the subscriber access-line is analog. If the user has ISDN or broadband access, he may set call-forwarding unconditional. The destination of call-forwarding is Gateway.

Once the Internet user has registered for VoIP, configured the call-forwarding service in subscriber database and Forwarding Service in Internet, he will become reachable by the caller, via PSTN or IP.

3.3 Incoming Call Scenario

To be reachable for either in Internet session or by PSTN phone under one single calling code, the called party needs only to set his call-forwarding service in the PSTN switch. There is no additional modifications in PSTN switch needed.

For a user using analog subscriber access-line, it is needed to set call-forwarding, for example, on busy.

When a call arrived, two situations are needed to consider:

- 1.If the called party is in an Internet session, the call is forwarded to the VoIP Gateway. After the Gateway receives the call, it requires the Gatekeeper to check whether the called party is in an Internet session and whether the called party has the right to access VoIP services. If both criteria met, the Gatekeeper responds a positive confirmation back to the Gateway. The Gateway then looks up the Forwarding Service (FS) for the new destination of the call. If the destination is a VoIP terminal, the Gateway can then establish a connection to the caller - the PSTN user. On the other side, the Gateway converts voice

signal into voice packages and send them to the diverted VoIP terminal (the called party's forwarding number) and vice versa. If the destination is PSTN calling code, a usual PSTN call connection will be established without Gateway participation.

2. If the called party is not in an Internet session, and phone line is busy, the call is also forwarded to the Gateway. Since there is no information of the called party in the Gatekeeper, the Gateway then should return a BUSY message back to the caller. Therefore the same state is to be delivered to the caller, the called party is busy.

3.4 Outgoing Call from IP-user Scenario

To make a call the Internet user must at first send a SETUP message to Gatekeeper.

The Gatekeeper then consults the Forwarding Service (FS) to see:

1. If the called party is in an Internet session and his VoIP terminal address is present the Forwarding Service (FS), the call can be directly forwarded to called party as VoIP call.
2. If there is no information of the called party, the call is forwarded to VoIP-Gateway assuming that called party is a PSTN user. Since all necessary verifications including authentication, access right etc. are completed in the registration phase, the setup process becomes more simple. Only minimal interactions between Gateway and Gatekeeper are needed. After a successful establishment of a connection to the called party - PSTN user, the Gateway converts the voice packages as PSTN signals and sends to PSTN user.

3.5 Forwarding Service

The Forwarding Service (FS) is used to store different types of call transfer destinations for a given Internet user / PSTN subscriber.

The call transfer destinations rely on the presence information stored in Presence Service (PS). The destinations can be either a VoIP terminal address, an e-mail address, or even another PSTN calling code.

■

If in FS the call is forwarded to a VoIP terminal, the Gateway will convert the incoming PSTN call into IP packages. If in FS the destination is Internet Call Waiting (ICW) client, the FS will act as ICW server and communicates to the ICW Client, running on the called party's computer.

ICW Client prompts a Dialog Box to the called party inquiring treatment for the pending call. The called party can select to send pending call to the voice mail, give it a busy signal, or give another PSTN calling code to complete the call, or even accept it by ending the current Internet connection, wait for a ring to answer the call.

■

For an e-mail address in the Forwarding Service, the incoming call is transferred as voice-file and is sent following the e-mail address.

■

For a usual PSTN calling code in Forwarding Service, the incoming call is transferred to the new PSTN call destination.

3.6 Web based OA&M service for IP-User

Setting call-forwarding service with different options on PSTN side is one of the OA&M (Operation, Administration and Maintenance) services provided by LEC to PSTN subscriber. A usual way is using the so called Subscriber Controlled Input(SCI) by dialing normal key-pad telephones. It is also possible to call the LEC operator. The operator sets the call-forwarding options from so-called OA&M terminal.

Because of roaming, it is impossible for a IP-user to set his call-forwarding service options from his phone. It is also difficult for a IP-user to call his LEC operator to set call-forwarding service. A better way is to allow the remote user himself to manage the call-forwarding. This can be done by the web-based OA&M service Subscriber Controlled Input via Internet (ISCI). After authentication, a remote IP-user can query the subscriber information and set call-forwarding on busy, or unconditional or no replay accordingly. The ISCI Service sends these settings to the PSTN switch.

The interface between ISCI and PSTN switch can be either MML(Man-Machine-Language) or Q3-Interface defined by ITU-T. Q3-Interface uses common management services and protocols (CMIS, CMIP) and provides a way for object oriented information exchange. Here, it is better to add an adapter between ISCI Service and OA&M of PSTN. On one side ISCI provides a vendors independent interface, on the other side the adapter provides a vendor dependent (proprietary) control model and interface.

In this way the system minimizes the vendor dependence. For the simplest configuration, all call-forwarding is managed by key-pad telephones or by the LEC's administration operator. This is vendor independent. For the advanced configuration, all call-forwarding is managed

by IP-user using the web-based Subscriber Controlled Input via Internet service. In both cases, we do not need to change any attributes in the subscriber database, what we change is the available attribute values.

3.7 New Supplementary Services

The new architecture of the VoIP for PSTN provides some new services for the VoIP user. For example,

Absent Subscriber Service,
Selective Call-forwarding, etc.

All these services are location independent.

Absent Subscriber Service is usually provided by the LEC. If, for example, the PSTN subscriber is not at home, when an incoming call arrives, LEC will turn on the voice mail, which is similar to answering machines, to take a message for the subscriber. With help of Presence Service and Subscriber Controlled Input via Internet, a remote new login IP-user can indicate the voice mail system to send the recorded voice as IP packages to computer. Thus Absent Subscriber Service together with VoIP services will prevent the user from missing messages from callers, even the user is far away from home.

Selective call-forwarding has two meanings: on PSTN switch side, it allows the Internet user to manage the call-forwarding settings by either ISCI browser or LEC operators. On the Internet side, it allows the Internet user to manage the Forwarding Service (FS). This provides a way to reroute the incoming call either to a VoIP terminal, to another PSTN phone, as voice mail, or to take the incoming call by dropping current Internet connection and waiting for a ring. The last three operations are controlled by Internet Call Waiting (ICW) Client.

4. CONCLUSIONS

In this paper, we present an architecture for Voice over IP services between PSTN and ISPs. Deploying this architecture, an Internet user can be called with his/her original telephone number. It also provides a way for call roaming.

This service profits the Internet users, the LECs and ISPs. This benefits IP users from not missing any important telephony calls during an Internet session. It benefits the PSTN carriers to generate revenues from both

telephony calls and Internet on-line access usage at the same time. It benefits the ISPs from providing the value-added service to increase their profitability.

5. ACKNOWLEDGMENTS

The author would like to acknowledgment Wilhelm Wimmreuter, Josef Schmitt, Dr.Redda Reda and Polly Yu for their useful comments, suggestions and improvements to this paper.

6. ABBREVIATIONS

CMIP	Common Management Information Services Protocols
CMIS	Common Management Information Services
FS	Forwarding Service
GK	Gatekeeper
GW	Gateway
ICW	Internet Call Waiting
IN	Intelligent Network
ISCI	Subscriber Controlled Input via Internet
ISP	Internet Service Providers (ISP)
LEC	Local Exchange Carriers
IETF	The Internet Engineering Task Force
ITU-T	The International Telecommunication Union, Telecommunication Standardization Sector
LS	Location Server
OA&M	Operation, Administration and Maintenance
PS	Presence Service
PSTN	Public Switched Telephone Network
RADIUS	Remote Authentication Dial In User Service
SIP	Session Initiation Protocol
SCN	Switched Communication Network
VoIP	Voice over IP

REFERENCES

- [1] S. Morgan, M.Delaney: The Internet and the Local Telephone Network: Conflicts and Opportunities. XVI World Telecom Congress Proceedings. 11.1997.

- [2] ITU-T Rec. H.323v2: Packet-based multimedia communications systems. 1998.
- [3] IETF Multiparty Multimedia Session Control (MMUSIC) working group and the Session Initiation Protocol (SIP). <http://www.ietf.org>
- [4] IETF PSTN and Internet Internetworking (PINT) working group.
<http://www.bell-labs.com/mailling-lists/pint/>
- [5] Dr. U.Schoen, etc.: Convergence between Public Switching and the Internet. XVI World Telecom Congress Proceedings. 11.1997.

INEXPENSIVE OPEN DISTRIBUTED SERVICE PLATFORM

Tor Martin Didriksen, Lars Sivert Sørungård, Tor Øystein Molnes
Telenor R&D, Kongens gt. 8, 7004 Trondheim, Norway

Key words: Intelligent Networks, Distributed Platforms, CORBA, Parlay

Abstract: This paper presents a platform to interconnect an IN node (SCP) to other communication networks (Internet) in order to introduce new IN services and extend capabilities of existing ones. The experimental platform presented is based on standard industrial hardware and software components. A hypothetical case study illustrates the versatility of the various interfaces provided by the platform.

1. INTRODUCTION

The telecom service providers tend to constantly offer more advanced services. Service uniqueness is a means of success in a competitive telecom market. The intention of intelligent networks (IN) is to provide means to meet these challenges by offering a platform for service development and execution. This goal has to some extent been met, and ongoing standardisation [1][2][3] indicates that IN tries to keep pace with the growing demands. Still, between one and three years are probably required for successful introduction of a quite new tailored service [4]. The main reason for this is that the SCP manufacturers have failed in providing platforms supporting *standard tools*, *multi-protocols*, and *standard databases*.

A serious problem caused by the lack of tools, and thus specialised skills required to developing IN services, is *poor programmer availability*. We believe that an open platform is the only means to decreasing *cost* and *time-to-market* for new IN services.

This paper describes the architecture of a distributed and reliable experimental service platform that is not intended for IN services exclusively. The platform is based on standard hardware and software components.

Section 2 describes a typical telecom service that will be referenced throughout the paper. Section 3 identifies a number of requirements to a service platform. Based on these requirements, Section 4 presents an experimental platform suitable for future services. Section 5 shows how the experimental platform can be opened up against additional access networks as well as external service providers.

2. CASE STUDY: TRAVEL AGENCY (TA)

This section describes a service for handling customers calling a travel agency (TA). We assume that the service was originally developed and available for PSTN only. While introducing the experimental platform later on, we will show how the service may be extended to include interfaces to additional access networks like *mobile networks* and the *Internet*.

We will also show how to outsource the execution and development of the service to an external service provider interfacing to the experimental platform using a standardised interface like *Parlay* [10].

The reader should note that the service described here is just a hypothetical example to illustrate the discussion in this paper.

2.1 Functional Properties of the Service

The TA has one toll free phone number, and a number of offices handling calls to this number. Each office has a number of travel agents. The caller is first exposed to a menu listing a number of options, e.g. *domestic travels*, *travels abroad*, *holiday travels*, etc. Each option is associated with a digit to be entered by the caller to indicate the topic of the request.

Each option in the menu is associated with a dedicated queue and a list of offices able to handle calls concerning this topic. Thus, an office is homogeneous in the sense that all agents in the office can handle the same type of requests. After selecting the most suitable option, the call is first attempted transferred to the offices in the list associated with the queue. If all agents in all relevant offices are busy, the call is inserted into the queue. Regularly, the first call in the queue is attempted transferred to the offices.

When the caller is eventually connected to an agent, the two parties agree on a product and a price. The caller may provide his credit card number to pay in advance, or he may choose to be invoiced. Information about regular

customers may be registered in advance and stored as customer profiles at the TA in a database.

2.2 Service Architecture

A possible overall architecture of the travel agency service is depicted in *Figure 1*. There are basically three independent parts in this architecture:

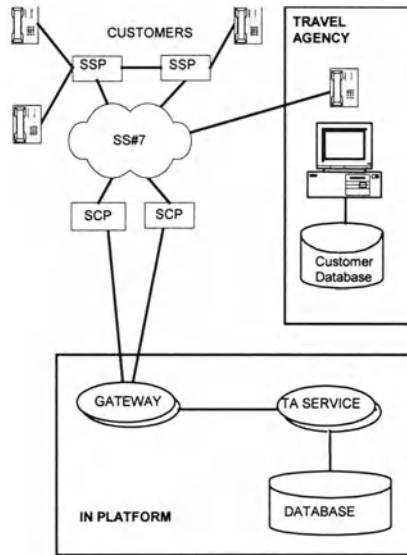


Figure 1. Hypothetical travel agency service

1. The customers, PSTN and the SCP.
2. The IN Platform with the Gateway to PSTN, the service for queue handling and a database to store the queue.
3. The travel agency, possibly with its own customer database.

The service described here is very basic, akin to what many companies have. Towards the end of this paper, we will explain how the service may be extended in various ways, with respect to functionality as well as how to outsource the service to an external service provider, and finally how to open up the platform for additional access networks.

3. EXPERIMENTAL PLATFORM REQUIREMENTS

An experimental platform for IN services should be:

- *Open*: In this context, this is related to the concept of using standard hardware and software as platform building blocks.
- *Reliable*: This is a very broad term [5] and includes both *availability* and *security*. Telecommunication services traditionally have a high level of reliability.
- *Distributed*: This means that the platform should facilitate the decomposition of a service into autonomous entities communicating through specific interfaces.
- *Accessible*: This means that the platform should be available in two respects: To end users, by means of various access networks, and to service providers (other than the network provider), by means of a standardised interface to the service platform.

In the following sections, we will elaborate on these requirements.

3.1 Openness

An open service platform means a platform implemented using widely applied, standard, commercially available hardware and software components. There are a number of reasons why we consider this a requirement to the platform:

- Applying widespread technology means that the number of alternative solutions to choose from is high, and that a suitable product in terms of cost and quality can be purchased.
- Hiring qualified and skilled people to develop and maintain the service platform and the services is easier since more people can be assumed to have experience in the technologies applied.
- Interfacing the platform to other entities such as additional access networks and services provided by external providers is easier.
- Keeping up to date with the technological evolution is easier because new technology is likely to be based on the most promising and widespread existing technology.

3.2 Reliability

The platform should be *fault tolerant*, i.e. be able to recover from failures without performing incorrect actions. *Recoverability* means that failed components are able to restart themselves and rejoin the system after the cause of failure has been repaired. There are two related aspects of reliability:

- **Availability**: The platform should provide *high* to *continuous availability*. This means that the platform should be able to resume providing services *during recovery* from failures. By minimising

recovery time, the platform is capable of providing virtually uninterrupted service to its users.

- **Security:** Security protects an information system from unauthorised attempts to access information or interfere with its operation [6]. Since the platform should be accessible by both *subscribers* and *external service providers*, a multitude of security issues should be investigated:
 - Identification and authentication of both humans and objects.
 - Authorisation and access control.
 - Confidentiality.
 - Security auditing.
 - Security of communication.
 - Accountability and Non-repudiation.

Accountability and *Non-repudiation* are related terms. Users are accountable for their security-relevant actions. Non-repudiation provides irrefutable evidence of actions such as proof of origin of data to the recipient, or proof of receipt of data to the sender to protect against subsequent attempts to falsely deny the receiving or sending of the data.

3.3 Distribution

Distributed object technology is well-suited for creating flexible systems because the data and business logic are encapsulated within objects, allowing them to be located anywhere within the distributed system. This is beneficial in terms of providing a logical decomposition of the system into independent units with clear interfaces, but also because the logical separation of entities easily can be extended to a physical distribution as well, thereby facilitating replication and increased reliability.

Another benefit of distribution is that scalability is supported since the system easily can be augmented with a number of additional components of a certain type. This of course assumes that there is a mechanism for distributing system load among numerous components of the same type.

Distributed objects separate their interfaces from the implementation. They are able to describe their interfaces, events and properties “on the fly”. The implementation language of server objects is transparent to clients, which enables the use of distributed objects as wrappers for existing applications regardless of implementation language. Existing IN services should therefore smoothly migrate to the new platform.

The basic idea of the distributed object middle-ware architecture is the object bus, which e.g. in CORBA is the *Object Request Broker* (ORB) that lets objects interoperate across address spaces, languages, operating systems and networks. The bus provides mechanisms that let objects exchange meta-

data and discover each other. At the next level, the infrastructure augments the bus with system-level services including licensing, security, version control, persistence and transactions. Thus, many important low-level services, previously implemented by means of proprietary mechanisms, are now available as parts of off-the-shelf commercial middle-ware systems.

3.4 Accessibility

The experimental platform should provide a standard interface for access by *External Service Providers* (ESP). The *Parlay Group* (AT&T, BT, Cegetel, Cisco, Ericsson, IBM, Lucent, Microsoft, Nortel Networks, Siemens and Ulticom, Inc) focus on the production of an API specification that enables enterprises outside of the network domain to access network information and control a range of network capabilities. The published Parlay API has quickly gained popularity for this purpose, and is briefly described below. It should be noted that there are other alternatives to such an interface, however, we choose to base our discussion on Parlay. *Figure 2* shows where the Parlay API fits into the architecture. The figure and the description below are extracted from [10].

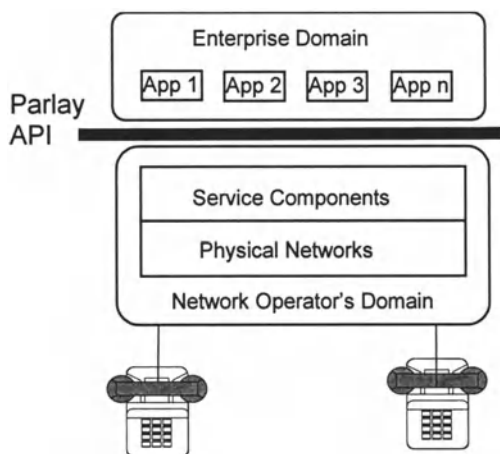


Figure 2. Parlay API

The Parlay API consists of two categories of interfaces:

1. **Service Interfaces.** These offer applications access to a range of network capabilities and information.
2. **Framework Interfaces.** These provide the supporting capabilities necessary for the Service Interfaces to be secure, resilient, located and managed.

Examples of the supporting functionality provided via the *framework interfaces* are: Authentication, discovery, event notification, integrity management, and finally operation, administration and maintenance.

Examples of *service interfaces* are: Generic call control service, INAP [1] call control service, generic messaging service, generic user interaction service, and call user interaction service (voice prompt to user, DTMF input from user).

A key to Parlay's success may be that the API is described using Microsoft and *CORBA IDL* [9]. This means that the API fits nicely into the overall requirements to the architecture of the experimental platform since standard open technologies are applied.

4. PLATFORM ARCHITECTURE

Based on the requirements identified in the previous section, we have developed an experimental platform for development of telecommunication services. *Figure 3* shows the experimental platform architecture. It interconnects through several interfaces to different kinds of access networks (currently SS#7, X.25, TCP/IP and CORBA). The platform is designed to be both *scalable* and *fault tolerant*. This chapter describes the platform architecture and the middle-ware software used to interconnect the components of the platform.

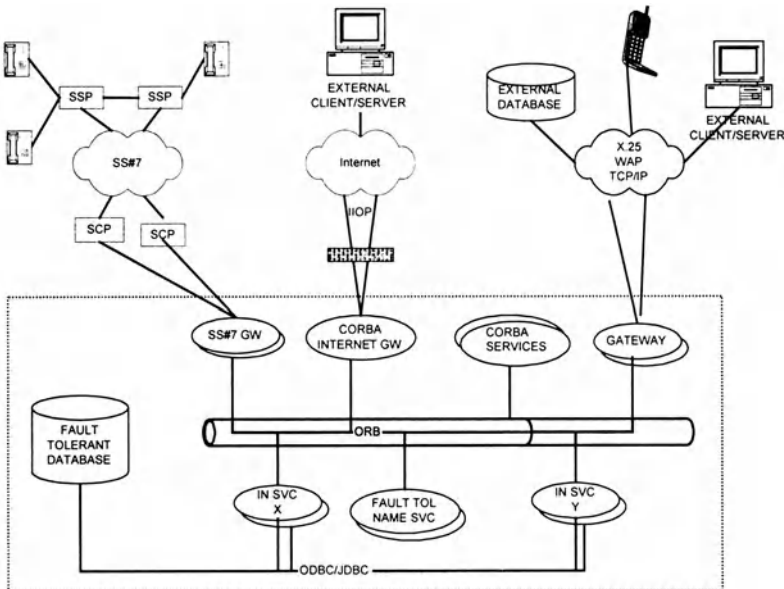


Figure 3. Experimental IN service execution platform

All items within the dashed line are part of the platform. Circles denote functional CORBA-aware units. Stacked symbols indicate replication, and replicas always run on separate physical hosts for fault tolerance and load sharing.

4.1 Overview

In *Figure 3*, gateways (GW) are generic (i.e. service independent) protocol converters, converting requests and responses to/from CORBA. Some of the gateways also have routing and firewall functionality, e.g. to overcome the Java “sandbox” limitations. The database is a standard SQL Server, but it is made fault tolerant by running it on an NT cluster. It is used by the services to store state between requests belonging to the same dialog. A fault tolerant name service uses “hot” replication based on group communication. Based on this platform we have implemented a variety of services, including:

- Traditional IN services (miscellaneous forms of call routing) with web interfaces for customisation.
- A service creation environment consisting of a generic set of building blocks for creating call centre solutions.
- An electronic commerce service, with external interfaces for customers, merchants and banks.

The functional units of the architecture of the platform are:

- **SCP:** The SCP implements a generic script which simply passes INAP requests to the SS#7 gateway. This design allows new service deployment without SCP MIB programming.
- **SS#7 gateway:** The CORBA-aware gateway converts SS#7 INAP requests to CORBA, analyses the message and hands it over to an instance of the appropriate replicated service objects. Upon reception of an answer from the servicing object, the reverse action is performed. Missing response from the requested object causes retransmission to another instance of the desired process, residing on another application server.
- **Middle-ware:** *CORBA* represents the platform middle-ware, i.e. the runtime environment or *Object Transaction Manager (OTM)* for our service implementation objects or *components*.
- **Application server:** A service object implements service specific logic. This facilitates the introduction of new services by deploying new service objects on the application servers.
- **Database server:** The internal database hosts data that needs to be stored during service execution, e.g. data about the state of service

progress. It can also be used to host data for a service that runs solely on the experimental platform, and therefore has no external database. The current implementation relies upon a Microsoft SQL Server 7.0 running on a fault tolerant NT Cluster.

- **Gateways:** Gateways are developed on demand. Of particular interest are gateways to:
 - The Internet.
 - Mobile terminals through WAP/WML.
 - External service providers.
- **Fault tolerant naming service:** The connections between Clients and Servers within the platform are permanent in order to avoid object binding for each service request. In the case of failure, some rebinding is required. The fault tolerant naming service enables rebinding of *all connections* upon failure detection instead of rebinding within a service request. This service is implemented by means of a replicated *supervisor* process. The supervisors implement the *Replica Control Protocol* [8] for group communication.
 - *Servant Objects* register to the replicated name service by providing *kind* (type) to identify redundant objects.
 - *Clients* request object *kind* in order to retrieve an IOR to any object of requested type, followed by an optional *register interest* message to the supervisor. Clients commit to reporting bad references to the name service, which in turn notifies all interested clients about the bad reference and provides a new IOR.

4.2 Satisfaction of Platform Requirements

Since we have already identified several requirements to a new platform for telecommunication services, it is of course relevant to discuss the proposed experimental platform according to these requirements.

- **Openness:** The platform is based on CORBA, which is a commercially available system for communication in a distributed environment. A standard language is used for defining object interfaces. CORBA is language independent.
- **Reliability:** CORBA provides mechanisms to handle application errors through e.g. distributed exception handling. Additional services to restart failing components may be implemented based on CORBA.
 - **Availability:** The fault tolerant naming service ensures that object references are valid, even in the case that the server fails. All servers are stateless. If a servant object needs to store state between invocations, it will use the (fault tolerant) database. Fault

tolerance will be easier to implement in the future as extended versions of CORBA will provide built-in support for this.

- **Security:** CORBA provides basic mechanisms for secure communication by means of Secure Sockets Layer (SSL). Additional CORBA security has been specified, but is not widely available yet.

- **Distribution:** CORBA provides the necessary means to define clear interfaces in a programming language independent manner. Thus, components of various implementation languages may communicate. Components on different ORBs may communicate through the Internet Inter-Orb Protocol (IIOP). Thus, physical distribution is well supported.

- **Accessibility:** Gateways may be implemented to provide interfaces to additional access networks as well as external service providers. The versatility of CORBA makes it possible to implement the gateways in a number of languages.

We feel that the last item concerning interfaces to the service platform is one of the most important aspects of a modern service platform. Thus, we discuss this topic further in the next section providing examples of how the travel agency service may be provided as an external service interfacing to the new service platform, and how the service may easily be extended to support additional access networks.

5. EXTERNAL INTERFACES

This section describes how adding various interfaces to the platform may extend the capabilities of the travel agency service introduced in Section 2.

5.1 Functional Extensions of the TA Service

A possible extension to the service is to add capabilities for online payment processing. Thus, a customer can choose to register himself with the travel agency by providing information about credit card numbers, bank accounts etc. This therefore becomes an extension to the previously existing customer profile.

After the conversation with the agent is finished, the call can be transferred back to the TA service, and a payment dialogue can be initiated. This assumes that there is a connection between the travel agency computer and the service, to let the TA service be notified about the payment details. An architecture supporting this functionality is depicted in *Figure 4*. Here, the TA service provides more functionality than the service introduced in

Section 2 since it also has to support processing of payment transactions and communication to external banking services etc.

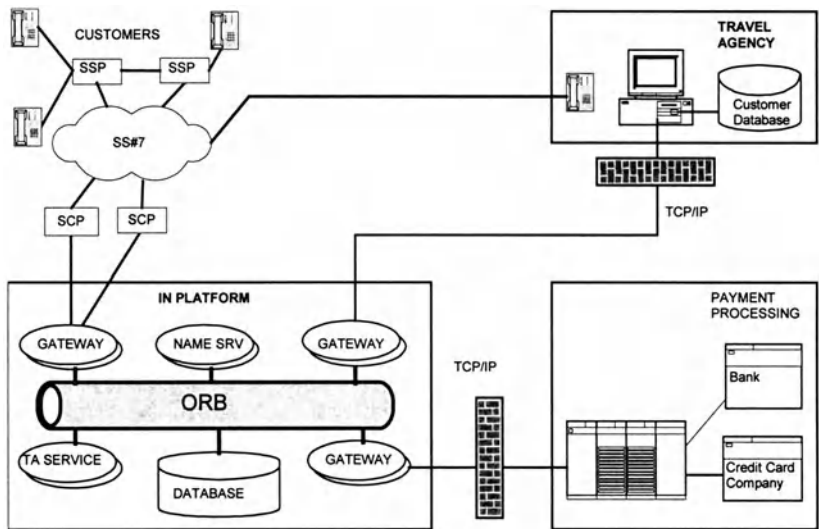


Figure 4. Extended functionality of the TA service

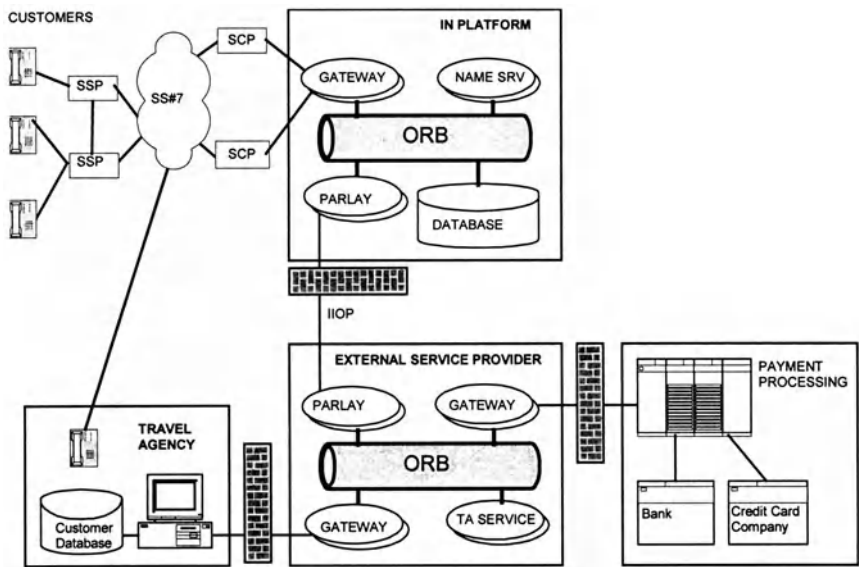


Figure 5. Parlay/IIOP Interface to external service providers

5.2 External Service Provider

The overall architecture for a service platform providing access to external service providers (ESP) is shown in *Figure 5*. The interface is built according to *Parlay* on top of CORBA, thus requiring a CORBA-enabled external platform.

The Parlay interface allows service requests to be passed to the external service provider. Similarly, the ESP is able and allowed to perform PSTN switching. The entire TA service may in this way be developed and maintained by the ESP. In the architecture above, we assume that the payment processing is also an external service with a clear and secure interface to client services. The payment processing service, however, does not need to interact directly with the service platform, and may therefore communicate with the caller through the TA service.

5.3 Gateways to Additional Access Networks

If we concentrate on the upper half of the architecture in *Figure 5*, we will see how the service platform can be extended with gateways to additional access networks. The important point here is that the presentation layer is separated out and made part of the service platform, while the service itself, which is still provided externally, remains unchanged. This separation of the service into different layers is similar to the approach often used in traditional software development projects, where an application is divided into three tiers: *User interface*, *business logic* and *database*.

To illustrate the addition of gateways to access networks, we can consider providing the service for Wireless Application Protocol (WAP) enabled mobile telephones. Adding a gateway to other networks such as the Internet would be quite similar.

The functionality of the service remains basically the same. When using a WAP phone, the customer places a call to the toll free number, and is exposed to a menu presented textually on the telephone. An option is chosen, and the call is placed in a queue in the same way as for PSTN. Then, the connection can be terminated, and the customer can wait offline. When the call is ready to be serviced, the customer is notified through WAP notification, and may choose to answer or abandon the call. If answering, the conversation with the agent takes place, and finally the payment transaction can be carried out using textual menus displayed on the telephone rather than voice messages. An architectural design supporting this solution is depicted in *Figure 6*.

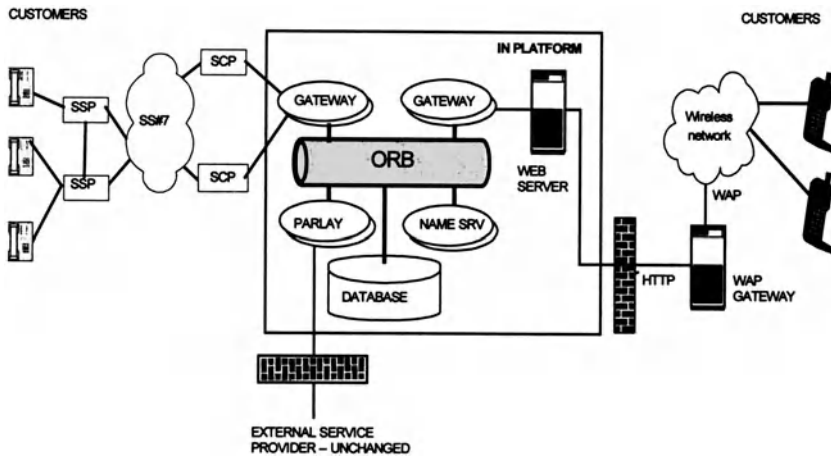


Figure 6. WAP Interface to mobile users

A web server is added to the service platform to run servlets providing the link between the WAP gateway and the CORBA objects of the service platform. The servlet generates output written in Wireless Markup Language (WML) which is eventually presented on the telephone. The WAP gateway is a protocol converter that converts WAP to HTTP. A firewall protects the service platform against unauthorised intrusion.

6. CONCLUSION

We have identified and argued that a new, experimental platform for the development of telecommunication services should be: *Open, reliable, distributed* and *accessible*. An experimental platform has been established according to these requirements, and initial experiences indicate that this is a step in the right direction. However, more experience is needed, particularly concerning security, before the platform can be used for supporting commercial services.

We have been experimenting with various services to identify common functionality to be considered part of the platform, and have thus implemented a fault tolerant naming service to handle failure of CORBA objects. A related service could be implemented to support error detection and –handling to provide a more flexible, generic and component-independent handling of failures. Report generation and logging could also be supported by this service.

Current efforts are being made to investigate an extension of the platform to include additional access networks, in our case a mobile network carrying

WAP. Our experiences so far are that the solution indicated using a servlet connection is flexible, simple and reliable. The hard part is to make the presentation layer generic to avoid tailor-made user interface components for each service.

In parallel, we are experimenting with external services interfacing to the service platform using Parlay. For traditional PSTN-based services, Parlay appears to provide the necessary functionality for call control, switching etc. The weakness is the lack of support for including service-specific interfaces, which is quite cumbersome at the moment.

Combining the two efforts, thus implementing an external service interfacing to a service platform providing access to multiple networks, revealed interesting problems. Parlay appears to be a rather cumbersome solution for services accessed by a network based on data traffic (i.e., not speech). In particular, online communication, which would be done through conversation using PSTN, is hard to achieve due to the missing support for service-specific interfaces in Parlay.

REFERENCES

- [1] European Telecommunication Standards Institute, Network Aspects 6: Intelligent Networks, www.etsi.fr, 1992
- [2] International Telecommunication Union – Standardization Section, Q.1200-series, Intelligent Network Recommendation, www.itu.ch, 1993
- [3] Telcordia (Bellcore), Advanced Intelligent Network Requirements, www.telcordia.com, 1992
- [4] H. Zuidweg, P. Quentin, G. Reyniers, E. Devleeschouwer, B. Quiryen: A Distributed CORBA-Based IN Architecture, 4th International Conference in Networks, Bordeaux, November 25-28, 1996
- [5] Kenneth P. Birman: “Building Secure and Reliable Network Applications”. Manning Publications Co. Greenwich, CT, 1996. ISBN 1-884777-29-5
- [6] Object Management Group, 1996, “CORBA Security Service Specification”.
- [7] Gray J. and Reuter A.: “Transaction Processing: Concepts and Techniques”. Morgan Kaufmann, 1992
- [8] AMR EL ABBADI, DALE SKEEN & FLAVIU CHRISTIAN, *An Efficient, fault-tolerant protocol for replicated data management*, ACM 0-89791-153-9/85/003/0215, 1985
- [9] PARLAY Industry Working Group: CORBA IDL. <http://parlay.msftlabs.com/>
- [10] PARLAY Industry Working Group: Parlay White Paper. <http://parlay.msftlabs.com/>

A MULTIAGENT SYSTEM ARCHITECTURE FOR QoS MANAGEMENT IN DISTRIBUTED MULTIMEDIA SYSTEMS

Mamadou Tadiou Kone and Tatsuo Nakajima

School of Information Science,

Japan Advanced Institute of Science and Technology

{mamadou,tatsuo}@jaist.ac.jp

Abstract The performance of computer systems relies on several factors. In particular, Quality of Service (QoS) research encompasses all the aspects of the management and delivery of a sound service. As fluctuations and degradations are inherent part of any heterogeneous system, QoS maintenance and adaptation become crucial parts of end-to-end QoS provision.

We propose in this paper a QoS adaptation scheme based on the *mobile agent technology*. We introduce the idea of *macro-adaptation* as the coarse-grained adjustments, and *micro-adaptation* that describes all the fine-grained corrective actions done on resources. Here agents are organized in a hierarchy along the components of the distributed multimedia system. Our idea in this model is to structure in an integrated and hierarchical fashion an adaptation strategy that span all the configurations made of components selected by a QoS manager. Computation is adapted to the available services or moved to other components if necessary. The interactions between the mobile agents and the components interfaces incorporate the adaptation mechanisms.

Keywords: Multiagents, Mobile Agents, Quality of Service (QoS), QoS management, Distributed Multimedia Systems.

1. INTRODUCTION

The improvement of general computing systems performance and services has long been a common goal for many research areas. Specially, modern computing systems need to assure end-users about the quality of the service being provided. Nowadays, a number of applications like Video-on-Demand (VoD), Desktop Video Conferencing, Distributed Col-

laborative Environments, Distant Learning are taking advantage of the new capabilities offered by performant networks and end-systems. However, these applications will not be able to provide satisfactory level of service if a suitable QoS provision scheme is not designed to sustain their activities.

Originally, the principles of Quality of Service (QoS) were applied to computer network communications. As QoS is gaining wide acceptance, its mechanisms are being applied to end-systems in distributed multimedia systems (DMS) as well. The concept of QoS is best described through the levels of service, a given provider can offer in order to sustain an activity. QoS-driven systems must deal appropriately with failure or changing conditions inherent to their environment in order to keep user confidence. As a result, **QoS adaptation** becomes a crucial part of the whole QoS management process.

We propose the **Mobile Agent** technology to support an efficient QoS adaptation scheme in distributed multimedia environments. Mobile agents are new and attractive alternatives to the traditional client-server computing model. In this scheme, computation and interactions are moved to the location of the resources. This paper is concerned with the adaptation of QoS in distributed multimedia systems (DMS) using the mobile agent technology. A number of works ([4], [7] and [8]) have addressed the issue of adaptation with interesting techniques. Here, we suggest that QoS negotiation, resource reservation and admission control should be planned wisely to prevent significant QoS violation. Our approach based on the flexibility provided by mobile agents aims to honor user requests transparently. Agents, organized in a hierarchy along the components in a distributed multimedia system (DMS), cooperatively maintain the required QoS.

Following this introduction, in section 2, we introduce mobile agents and explain the rationale behind our choice in QoS adaptation. In section 3, we give a description of the mechanisms of QoS provision for distributed multimedia systems. We present in section 4 some noteworthy approaches to QoS adaptation and draw some comparison lines. Our integrated and hierarchical QoS adaptation strategy through the mobile agent technology is explained in section 5. Finally, we state concluding remarks and hints to further directions of research in section 6.

2. AGENTS: WHERE THEY FIT

2.1 Nature of Agents

There are two perspectives in defining the word *agent*. There is the software engineering perspective and the cognitive science perspective.

The first refers to a piece of software that can migrate autonomously inside a network while the second defines an entity or a set of entities called *multiagent system* that has capabilities, provides services and can act on their environment.

2.2 Mobile agents in DMS

A mobile agent is a program that can migrate autonomously across a heterogeneous network and perform tasks on behalf of its owner as shown in figure 1. At anytime it can halt, move together with its state and data to a new machine, and resume execution at the point where it stopped. This ability of agents to migrate from host to host makes mobility support a fundamental requirement for agents systems. In addition, the introduction of mobile code in local or open network like the internet raises several security issues. Agent and host authentication, authorization, portability and access control are some of these issues.

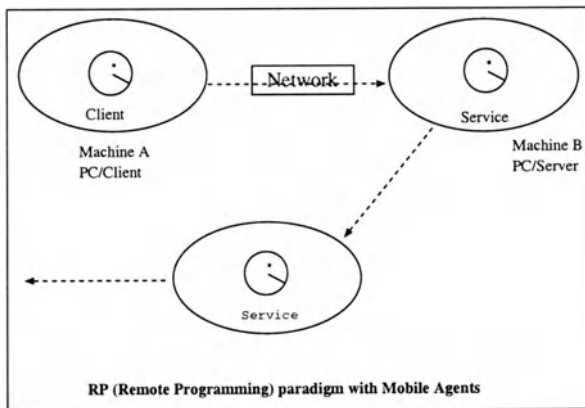


Figure 1 A new approach to client/server communications

Mobile agents are a new kind of abstraction in the client/server communications world. The mobile agent technology has found many interesting applications in areas where QoS control and management is an important issue like in [10].

In distributed multimedia environments, a component is defined as a physical (peripheral, network node, CPU ...) or processing entity with the ability to produce, transform or consume a multimedia data. A component as shown in figure 2, is characterized by its interface and services it provides. In addition to describing interaction requirements, behaviors and capabilities, interfaces present the set of services a component has to offer.

In place of the previous concepts of object oriented programming, we

use multiagents and mobile agents. With mobile agents come the possibility to move the interaction point, the service access point and computation to the target component side. An agent uses a service by invoking the component interface.

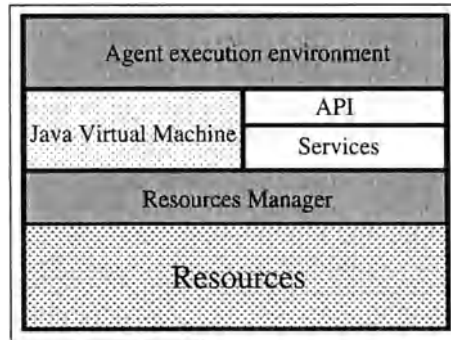


Figure 2 Anatomy of a component in a DMS

2.3 Multiagents systems

Multiagent systems - usually studied in AI - are distributed computing systems composed of several interacting computational entities called agents. These constituent agents have capabilities, provide services, can perceive and act on their environment. In our model, distributed multimedia components are treated as this type of agent.

2.4 Mobile agents and Fault Tolerance

As mobile agents are bound to perform many operations and use services provided at multiple hosts. Some concerns naturally arise about their reliability, their execution environment, the host machine and the network that support them. To deal with these problems we propose a simple scheme for failure recovery. At the time it leaves a host, an agent creates a checkpoint on a persistent store. On regular intervals, the replicated agent sends a message (ping) to check if the remote agent is alive. This way, when the current host crashes together with an agent or a communication failure occurs, the replicated agent at the last host visited recreates the previous state and carry on the task.

3. QUALITY OF SERVICE (QOS)

3.1 QoS Concepts and Management

In a distributed multimedia environment, any activity that contributes to the production, delivery and consumption of some multimedia data can be characterized by the quality of its service. The followings are some well accepted phases of Quality of Service provision borrowed from the network communications area that are used in distributed multimedia environments:

QoS Specification: A service user must specify to the provider with objective and subjective QoS parameters the level of service that it expects.

QoS Negotiation and Mapping: In this process, the system must identify the set of components that are able to provide and sustain completely the required level of service. Then, with adequate mapping, the required QoS are translated from one system layer to another.

Resource Allocation and Admission Control: In conjunction with the negotiation phase, necessary resources are reserved for an activity. The required resources availability could be an issue unless the admission control functions compares beforehand the activity needs with the existing resources. Admission is then granted when enough resources exist.

Monitoring: Once an activity has started, a given system layer constantly monitors the lower layer to check for alteration in the value of the performance parameters.

Maintenance and Adaptation: These processes aim to counteract the perceived degradation or violation of the contracted QoS. Adaptation is achieved through change in the topology of the components selected by the negotiation process, a coarsened or fine grained resource adjustment or an explicit change of service by the user.

Re-negotiation: When a degradation of the QoS level has reached a critical level and nothing further can be done to recover or adapt to a lower level of service, a re-negotiation is initiated by the system. This scheme proposes an entirely new QoS provision process. A change of service preferences by a user also can trigger a re-negotiation. The whole management process is depicted in figure 3.

In addition, the *QOS contract* binds the entities (network provider, network and end-system resources ...) involved in the QoS provision. We define it in generic terms as follows:

```
typedef struct{
    service-spec_t : service-spec ;
    committment_t : committment ;
```

```

monitoring_t : monitoring ;
adaptation_t : adaptability ;
dependability_t : dependability ;    }service-contract_t

```

The contract is composed of the *service specification*, the class of *commitment* the resources are able to provide to sustain a given activity, the *monitoring* process, the *adaptation* process, and a subjective parameter we call *dependability*.

A user states with objective and subjective parameters in the service specification the performance expected. For instance, in dealing with timeliness parameters like loss, delay, jitter and throughput could be considered.

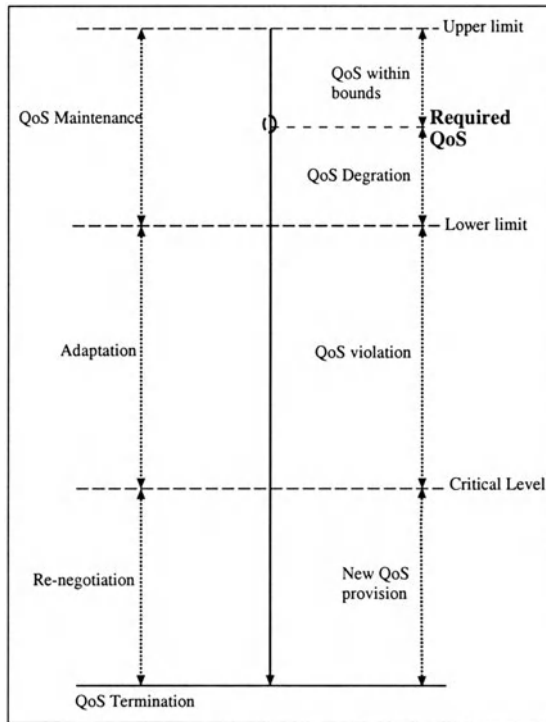


Figure 3 A simple view of QoS Management

The commitment clauses are known as **Deterministic**, **Statistical** and **Best effort**. The first guarantees that resources are exclusively dedicated to an activity. The second is a loose version of the first where a service tolerates some fluctuations because resources are shared and eventually pre-empted. The best effort commitment does not guarantee any level of service nor does it permit control over any resource. These

clauses are qualitative expressions that helps precise the nature of the service required.

4. A NEW ADAPTATION SCHEME

In a distributed multimedia environment, unstability, fluctuations in the level of service or temporary failures are inherent parts of the activity of any system. As a result, service maintenance or adaptation are required if user confidence is important. In our context some particular motivations are the following:

1. In distributed systems environments, shared resources due to system policies (resources management, priorities, ...) may not be available on request.
2. Congestions in the network or end-systems often occur.
3. Different clients QoS requirements, depending on preferences, may create conflicts.
4. Granted resources may be pre-empted and not be guaranteed over time.

In our approach to adaptation, we define **macro-adaptation** as all the coarse-grained adjustments relying on agents movement with change in components topology; whereas **micro-adaptation** describes all the fine-grained corrective actions done at the lower level resources along agents hierarchy. In addition, in this agent-centric design, we assume that QoS monitoring and resource reservation are done at the component level. Our idea is to organize an adaptation strategy where mobile agents move computation - like in figure 6 to components where services are available.

4.1 Agent-based adaptation strategy

How do mobile agent identify their destination before migration? Our scheme models components as a multiagent systems whose constituents can communicate in an appropriate agent communication language like the Knowledge Query and Manipulation Language (KQML). Before the start of a session, all agents (components) in the multiagent systems register their capabilities to a facilitator agent. Then, when a component agent fails to meet its requirements, it sends a message to the facilitator and ask for another agent component that is able to satisfy these requirements. When a suitable component is identified this way, a mobile agent can carry on an appropriate adaptation scheme.

An integrated approach:

The system we propose is integrated in two aspects:

First, all components spanned by related mobile agents, are integrated

in a single end-to-end QoS provision process.

Traditionally, the negotiation process identify a finite set of components that are able to sustain an activity and assign to each, in a rigid configuration, its share in the overall end-to-end QoS provision. In contrast, with the introduction of mobile agents, several components assembled in a loose configuration, are connected dynamically to provide the required QoS. Such an approach is described in [1]. In our scheme, system agents coordinate the operations of system components and communicate with network agents responsible of network transactions at the transport layer of the network as illustrated by figure 4.

Second, all adaptation mechanisms in research related to QoS provision have been implemented as isolated processes activated when a degradation or violation actually occurs. However, unstability, degradation are unavoidable events in heterogeneous environments. The QoS negotiation process, the resource allocation and admission control should incorporate strategies that prevent as much as possible violations of QoS requirements.

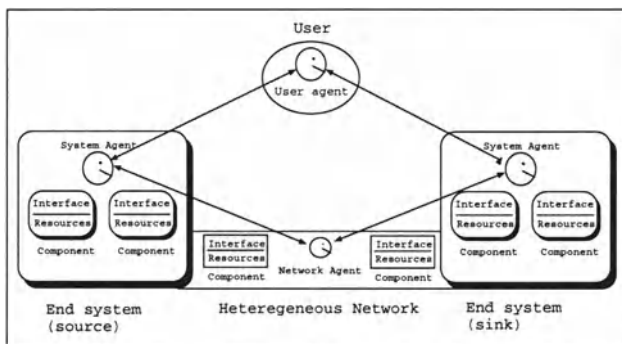


Figure 4 Agents structure in end-to-end QoS provision

A hierarchical approach:

When a degradation or violation is detected, maintenance or adaptation is carried out either locally or with the support of a remote component along the following steps:

1. The system identifies the component that fails to satisfy the local QoS.
2. The related agent (ex. end-system agent) attempts to solve the problem locally by either:
 - ◊ moving the computation to a peer component (ex. end-system component);
 - ◊ sending a message to a sub-component agent to make the necessary corrections down into the hierarchy or interact with the resource

manager.

◇ sending a message to another agent located elsewhere and the previous steps are performed there.

3. If local adaptation fails, the adaptation process is moved to another part of the system (ex. network) and step 2 is re-enacted.

These steps may be conducted in two ways:

Horizontally, computation is moved with the QoS access point from the current component to a peer component.

Vertically, a message is sent to an agent at a sub-component down in the hierarchy. The target agent can further move the computation to a peer sub-component if necessary. Should the QoS maintenance or adaptation to a lower QoS fail, the end-system agent sends a message to the network agent to carry out a similar strategy.

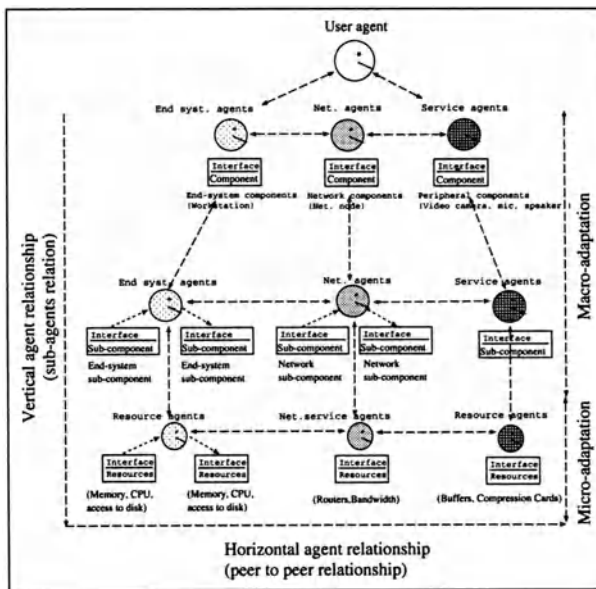


Figure 5 Agent-based hierarchical adaptation

4.2 Agent-based adaptation mechanisms

Agents interacting with components

The main purpose of our system is to make existing services at a component available on request to agents. In normal situation, the level of the service offered matches the request. Otherwise, a maintenance or adaptation to a lower level is proposed. Upon receiving from the user agent the QoS contract with the specified QoS parameters, an end-

system agent moves to the *agent execution environment* of a component as depicted in figure 5. In this environment, the agent makes the necessary computations, calls the *Application Programmer Interface (API)* in order to use the services.

An example of interface is the Java Database Connectivity Layer and the Java Advanced Windowing Toolkit. Here we assume that during the previous Resource Allocation and Admission Control, resources necessary to sustain the computation have been reserved. When there is a degradation and the level of service is not satisfactory and all attempts to maintain it has failed, the agent has the option to adapt its computation to the available resources. That is, the mobile agent alters a QoS level to a lower level.

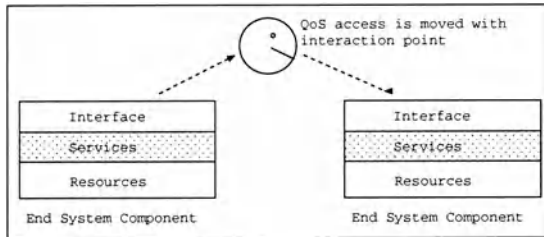


Figure 6 Agent-based Adaptation Model

Agents migration:

Agents mobility is supported by the object serialization functions included in the Java language. For the three types of agents (user agent, end-system agent and network agent) migration happens inside their respective domain (end system or network). An agent migrates between peer components of the same domain. A network agent assigned to nodes can migrate only between nodes and so do end-system agents. Often, the operation of a component depends on other sub-components. These sub-components are visited by related agents down in the hierarchy. All computations are identical at this level as at the upper level. At the bottom of the hierarchy, in connection with the resource manager, resource agents participate in fine grained adjustments called micro-adaptation by travelling between locations. An adaptation scenario is explained in table 1.1.

Inter-agents interactions:

Along the hierarchy, mobile agents send messages in a serialized form horizontally to peer agents located in another domain or vertically to agents related to sub-components or resources. The two types of communication are at the heart of agents cooperation for maintenance or adaptation.

QoS Adaptation Strategy	Context	Agent Migration	Agent Interaction
Macro-adaptation	End-system	WS ----> WS NSAP ----> NSAP	Component API Message
	Network	Node ----> Node Router ----> Router	Component API Message
	Peripheral (Video camera, Mic.)	Video Cards ----> Video Cards	Component API Message
Micro-adaptation	End-system	CPU/Memory ----> CPU/ Memory	Resource Manager Message
	Network	Buffer ----> Buffer	Resource Manager Message
	Peripheral	Codec ----> Codec	Resource Manager Message

Table 1.1 Agent-based adaptation scenario

4.3 An example

Let's consider a system that captures images and stores them in a remote multimedia database storage (MMDBS) as illustrated in figure 7. In addition, this data in the remote server is accessible to other users ready to be displayed locally. That is, the whole system can be used in two modes. When a user captures data and stores it remotely. The source is a camera together with a set of workstations linked in a sub-network and the sink is the multimedia database server. Conversely, a user in need of some multimedia data could fetch it from the database server and display it locally.

In the first mode, a QoS manager in its negotiation phase would simply select a configuration made of the following components: the camera, a workstation with a video capture card, a path of nodes and routers in the network and the database storage.

When all the resources allocation and admission tests complete, the service starts. We assume all the workstations in the subnetwork have the same capabilities and resources. The camera physically linked to a given workstation capture images which are compressed, sent across the network and stored in the database.

In the second mode depicted in figure 8, data is fetched from the remote server. The QoS manager selects a configuration made of the following components: the database server (the source), a path of nodes and routers in the network and a workstation (display sink). In fact, during the QoS negotiation phase, several potential configurations of components and resources were identified and recorded by the QoS manager.

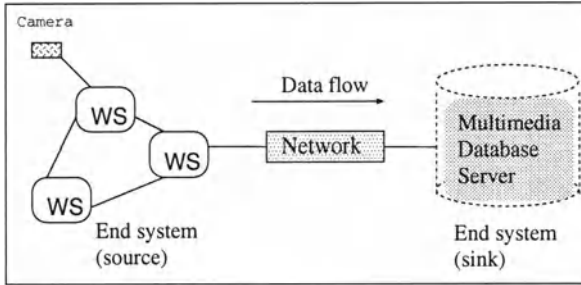


Figure 7 Example system: storing data in a multimedia database

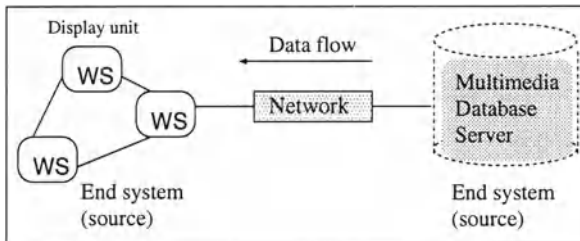


Figure 8 Example system: fetching data from a multimedia database

Here the user agent, system agents and network agents are located respectively in the user application, the end-system and the network. When a degradation or violation occurs, the QoS manager tries to locate and identifies the failing component (source workstation, network node, router or database server). The mobile agent located at the failing component has two alternatives. On one hand, it can send a message to other agents located at sub-components like the operating system, the hard disk, the memory or video capture card. On the other hand, the mobile agent can move the necessary data to another workstation (macro-adaptation) and restart computation where it stopped. As a result, the initial QoS level is maintained or adapted to a lower level. In case violation is detection at network node or router, the corresponding agent migrates with all the data to another node or router. Although the service started with a given configuration of system components, the mobile agents give us the possibility to move computation from one component to another. The configuration changes dynamically to cope with violations.

5. RELATED WORK

A number of works have addressed the problem of QoS adaptation with various techniques in different environments. Few of these ap-

proaches use the agent technology to improve QoS provision.

In [8], a QoS manager conducts QoS adaptation by interacting with static QoS agents bound to components. The QoS adaptation strategy relies on three schemes are. The first called Component Reconfiguration Scheme (CRS) performs adaptation by altering the components topology and initiating a smooth and transparent transition. The second named Resource Reconfiguratio Scheme (RRS), redistributes the level of QoS supported by each component in a given configuration to cope with violation and keep the overall end-to-end QoS.

In [6], the concept of binding as abstraction of communications between components is used to sustain the required QoS through monitoring, adaptation and reconfiguration. Here binding objects, organized in a hierarchy, apply adaptation policies dynamically.

In [4], a hierarchical QoS adaptation based on the organization of binding objects is proposed. In this paper, adaptation is carried out along a hierarchy of bindings. From the top with coarse-grained actions to the bottom with fine-grained atomic resource control, the adaptation mechanisms are embedded in the binding components which are conceived as communications abstractions. In addition, these bindings are responsible for the end-to-end QoS maintenance.

In [10], an agent-based approach is introduced to carry out corrective QoS adaptation in a cloud of non-RSVP-capable network routers. Static agents monitor tunnel properties and interact with RSVP routers to deliver the required QoS.

In the above systems, when agents are used, their functions are limited to monitoring the level of QoS and eventually notifying the system of QoS violations at the host component. Our approach that consists of moving the QoS access point from one component to a peer component with mobile agents is new. The initial configuration need not be altered unless no component is able to compensate the degradation. Adaptation is performed smoothly by mobile agents horizontally or vertically in an integrated fashion. System agents - at the source or sink - and network agents cooperatively manage the end-to-end QoS transparently.

6. CONCLUSION

We have presented a new approach to QoS adaptation that relies on the flexibility of mobile agents. In general QoS provision, a static configuration of system components is provided. To this static configuration, the mobile agent technology allowed us to substitute a dynamic one. The adaptation strategy evolves around the interactions of agents and components interfaces on one hand, and the cooperation between agents

themselves on the other hand. To achieve this, we have introduced macro-adaptation and micro-adaptation to describe the operations of agents that cooperate along a hierarchy. Although designing an adaptation strategy that responds to QoS violation might be desirable, planing wisely a negotiation, resources allocation and admission processes still remain the best way to prevent QoS degradation.

Our model - due to its integrated and hierarchical aspect - could improve the overall QoS provision significantly provided the entire system is reliable. We are planing to include the strategy described in this paper in a complete QoS provision scheme based on mobile agent in a near future.

References

- [1] Mamadou Tadiou Kone and Tatsuo Nakajima, "An Architecture for a QoS-based Mobile Agent System." In *Proceedings of 5th Int. Conf. on Real-Time Computing Systems and Applications*, Hiroshima, Japan, Sept. 1998.
- [2] Hutchison D., Coulson G. , Campbell A. , Blair G., "Quality of Service Management in Ditributed Systems." In Sloman M.(ed), *Network and Distributed Systems Management*, Chapter 11, Addison Wesley. Reading, Mass.
- [3] C. Aurrecochea, A. Campbell, and L.Hauw, "A Survey of QoS Architectures." In *ACM Multimedia Systems Journal, Special Issue on QoS Architectures*, 1997.
- [4] Daniel G. Waddington and David Hutchison, "End-to-end QoS Provisioning through Resource Adaptation." In *Proceedings of the 8th IFIP Conference on High Performance Networking - HPN'98*, Vienna University of Technology, Vienna, Austria, September 1998.
- [5] D. Waddington, Chritopher Edwards and D. Hutchison "Resource Management for Distributed Multimedia Applications." In *Proceedings of the Second European Conference on Multimedia Applications, Services and Techniques - ECMAST*, Milan, Italy, May 1997; LNCS 1242.
- [6] D.G. Waddington and G. Coulson, "A Distributed Multimedia Component Architecture." In *Proceeding of the first International on EnterpriseDistributed Computing*, Gold Coast, Australia, October 1997.
- [7] S. Fischer, A. Hafid,G., Bochmann,and H. de Meer, "Cooperative QoS Management for Multimedia Applications." In "4th IEEE Int. Conf. on Multimedia Computing and Syst.,"Ottawa, Canada, June 1997, pp. 303-310.
- [8] H. Hafid and G. V. Bochmann, "Quality of Service Adaptation in Distributed Multimedia Applications." In *ACM Multimedia Systems Journal, Vol.6*, 1997.
- [9] Luiz A.G. Oliveira, Paulo C. Oliveira and Eleri Cardozo. "An Agent-Based Approach for Quality of Service Negotiation and Management in distributed systems." In *Mobile Agents, Proceedings of the First International Workshop,MA '97*, Berlin, Germany, April 1997, pp 1-12.
- [10] Hermann de Meer, J-P. Richer, A. Puliafito and O. Tomarchio, "Tunnel Agents for Enhanced Internet QoS." In *IEEE Concurrency*, April-June,1998. pp. 30-39.
- [11] The Object Management Group (OMG), "The Common Object Request Broker: Architecture and Specification." OMG Doc.,<http://www.omg.org/>, Dec. 1995.

MOBILE AGENTS FOR MOBILE USERS

¹ Do van Thanh, ² Sverre Steensen & ³ Jan A. Audestad

¹ Ericsson Norway Applied Research Center, P.O Box 34 N1361 Billingstad, Norway -
etodvt@eto.ericsson.se - Tel: +47 66 84 12 00

² University of Oslo - UniK P.O Box 70 N-2007 Kjeller Norway -
sverrest@ifi.uio.no - Tel: +47 63 81 45 70

³ Telenor AS P.O Box 6701 St. Olavs plass N-0130 Oslo Norway –
jan-arild.audestad@telenor.com – Tel: +47 22 77 99 52r

Key words: Mobile Agents, Mobile code, Mobility, Distributed Object Computing, Open Distributed Computing, Mobile Computing, Telecom and Computing convergence

Abstract: In this paper we propose the use of mobile agents in the implementation of mobility support. The paper starts with a presentation of the mobile agent concept. Then, the notion of ideal user mobility is presented and followed by the current mobility support approaches. The three mechanisms used to support mobility, namely channel establishment, pre-duplication and dynamic duplication are then explained. It is then shown that the mobile agent concept is probably the most suitable concept to realize mobility support since it allows a dynamic combination of all the mentioned mechanisms. The paper concludes with some comments regarding the use of mobile agent concept and suggestions for future studies.

Background: The work described in this paper results from the mobility part of the TELEcom REsearch Programme TELEREP, carried out at the Ericsson Norway Applied Research Center (NorARC) in collaboration with the University of Oslo, Center for Technology at Kjeller UniK [1]. The programme has been established to obtain practical experience with Distributed Object Computing methods such as ODP/TINA [2] [3], [4], CORBA [5], mobile codes in the implementation of TMN and IN functionality, and to study how mobility and security could be provided in the DPE environment.

1. INTRODUCTION

The traditional approach to the specification and design of telecommunications services based on proprietary solutions and with low level of modularity does not meet the new needs. New technologies are required to facilitate the introduction of new services or the adaptation of the current ones. Fortunately, the convergence in technology enables the use of concepts, principles and methodologies from the computing world. It is therefore not surprising that concepts, principles and platforms defined in Open Distributed Processing (ODP) and Distributed Object Computing are adopted and used in the telecommunications systems, bearing in mind that these systems are probably the largest and most complex of all existing distributed systems. The main idea is to consider the distributed system consisting of heterogeneous nodes as a huge virtual mainframe. All the functions and mechanisms necessary to support distribution are hidden to the applications by the introduction of distribution transparencies. The defined transparencies are access, location, migration, relocation, replication, persistence, transaction and failure. An open distributed platform must at least offer access and location transparencies such as in the case of CORBA or Java.

However, there is a different concept called **mobile agent**, which does not provide location transparency to the applications. Indeed, the applications should exploit their location awareness to improve the performance. Mobile agent is actually not a new concept but is gaining more and more popularity with the growth of the Internet since the concept is intended for huge distributed systems such as the Internet, although it is not yet used on that scale. Although Open distributed systems provide process or object migration, it is designed with smaller scale computer networks in mind and thus assuming high bandwidth, small predictable latency and trust. The mobile agent concept addresses instead large scale setting where networks consist of heterogeneous hosts managed by different authorities with different level of trust, and connected by links with different bandwidths ranging from limited wireless connections to high speed optical links. This paper presents the approach of using the mobile agent concept in the design and implementation of of mobility support.

2. MOBILE AGENT CONCEPT

It is not trivial to provide an unambiguous and unanimous definition of an agent. We choose to adopt the basic agent model defined by [6]:

"A computational entity, which acts on behalf of others, is autonomous, pro-active and reactive, and exhibits the capability to learn, cooperate and move"

Autonomy implies that an agent should be able to operate independently from its user. Hence, an agent should be able to make rational decisions based upon the information it has gathered and the knowledge of its own task and purpose. **Pro-activeness** implies that an agent should take initiative in order to achieve its goal. **Reactivity** is the ability to sense and respond to changes in the environment.

We define a mobile agent as *a software agent that can move between locations in a heterogeneous network*. In addition to the basic model, a mobile agent is characterized by:

- a **life cycle model** defining the framework to create, destroy, start, suspend, stop dispatch and retract agents.
- a **computational model** referring to the computational capabilities of an agent, which include data manipulation and thread control primitives.
- a **security model** limits the ways an agent's ability to utilize the network and vice versa.
- a **communication model** focus on agent to agent communication and communication between an agent and e.g. the network. This can be based on simple message passing scheme or more complex approaches like KQML (Knowledge Query Management Language) or ACL (Agent Communication Language) proposed by FIPA (Foundation for Intelligent Physical Agents).
- and a **navigation model** gives the requirements to the issues of transporting an agent between two computational entities and of the naming and identification of agents in the system.

In order to build applications based on mobile agents that can be used on a large scale, the system must comprise a mobile agent framework, which supports all the mentioned models. An application is modeled as a collection of agents, each agent occupying different places at different times since it can move from one place to another. There is a diversion in framework concept. Some frameworks such as Aglet, abstracts everything in the system by the unique concept of agent. Some others such as Voyager allow a combination of agent and object concepts. Indeed, agents are viewed as extension of traditional objects. While the first approach yields a more coherent system that is easier to maintain and upgrade, the second approach enables easy adoption since migration from existing object system to agent system can be done gradually.

It is natural that the introduction of mobile agent framework will incur overheads in processing, memory demands, access response and execution time, etc. However, thanks to rapid and considerable progress of the

underlying technology the mentioned overheads are no longer insurmountable obstacles. Indeed, the most commonly used technology Java can be quite small and fit for embedded systems. In addition, the future arrival of Java chips and Java system extension like Jini will allow full deployment of agent based applications with acceptable performance.

3. IDEAL SITUATION FOR A MOBILE USER

To illustrate the feasibility of the mobile agent concept we are going to investigate how a strategically mobile agent may benefit a mobile user. When a user moves to a foreign host and after authentication and authorization he would often like to access the system according to his own user profile, i.e. having the same documents and applications, the same icons, the same look and feel, the same environment as at his home site.

Ideally a user should be able to access the same applications/services and all his files or documents in exactly the same way, in the same presentation, with the same look-and-feel that he has personalized, no matter when and where he is or no matter what terminal he is using. Indeed, ideally the technology should be able to adapt to the user instead of requiring that the user must adapt to it, as the situation is today.

As depicted in *Figure 1*, ideal user mobility means the ubiquitous availability of the user's applications, his data files and his user profiles independently of the terminal in use. The **applications** comprise both the communication ones such as voice phone, video phone, conferencing, etc and the computing ones such as word processor, spreadsheet, mail, specific software, etc. The applications can use one medium or multiple media such as voice, video and data. The **data files** are of all types, either structured documents or unstructured raw data files. They can contain data for different types of media such as text, sound, picture or video. The **user profiles** capture the preferences, subscribed services and personalization that the user has chosen. Each user may have several profiles, for example one main profile for the main presentation and one for each application.

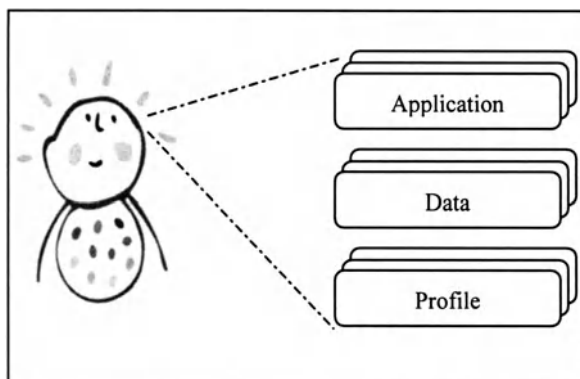


Figure 1. Ubiquitous availability of applications, data files and user profiles

However, the physical characteristics of the underlying network and the terminal such as display size, processing and storage capability, media type, etc. are the unalterable limitations that prevent the total realization of the described ideal situation. Fortunately, the user usually understands such limitations and he is willing to accept some degradation or alteration. But, he has more difficulty to accept that given the required connectivity and a terminal with the required capability he is not able to access his applications, his documents. It is neither acceptable to require that the user must have quite a lot technical knowledge or must perform cumbersome, and time consuming configuration tasks.

In order to have a clearer view on how mobility is supported today, let us consider successively the cases of GSM (Global System for Mobile communication), UPT (Universal Personal Telecommunication), Mobile IP and Telnet/Remote login.

4. GSM

GSM (Global System for Mobile communication) offers terminal mobility meaning that the mobile user get the access to his services through his mobile cellular phone. GSM has telephony, i.e. voice communication as main application or basic service and a few other applications called supplementary services. The applications/services are well defined and complies with the ETSI (European Telecommunications Standards Institute) specifications. The application software (also called logic) must be installed and run on every location that supports telephony and supplementary applications. Each user has a user profile containing service restriction data such as list of subscribed services, roaming restriction, security data,

charging data and routing data. The user profile is stored at the home site of the user, or more specifically in the Home Location Register (HLR). A copy is made and temporally stored at the domain currently visited by the user, or more specifically the Visitor Location Register (VLR). GSM offers very limited storage capability for user data. The existing user data are voice messages or SMS (Short Message Service) messages, which are saved in the home domain.

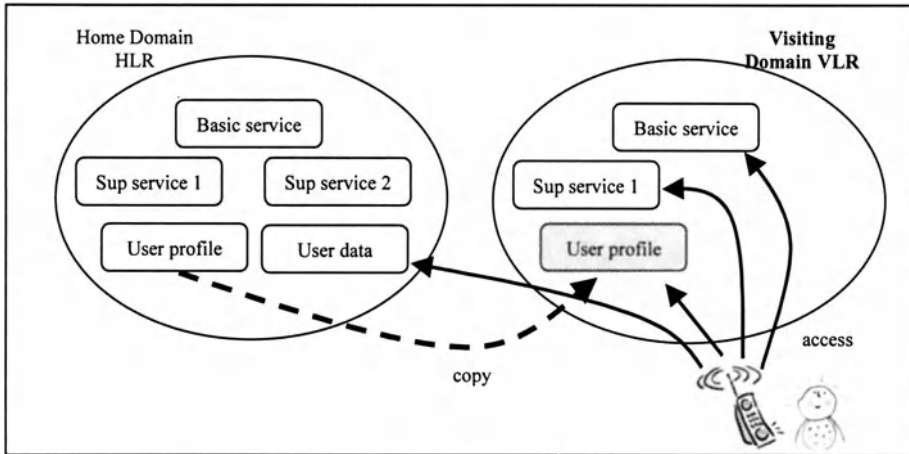


Figure 2. Mobility support in GSM

Figure 2 shows a user moving from his home domain (HLR) to a visiting domain (VLR) with his phone. In the visiting domain there is installed only logic for basic service, i.e. telephony and supplementary service 1 but not supplementary service 2. Supplementary service 2 is hence not available for him during his visit at the visiting domain. His profile is copied over to the visiting domain and made available for him. His data remains at his home domain and can be accessed through a channel between the visiting domain and his home domain. GSM is functioning very well today when the number and type of applications are relatively small. But in the future with the blooming of applications in number and type combined with the variety in user's preference and choice, it is not rational to expect that all applications will be present at every visiting domain. The user will as a result not be able to access all his applications. Another problem is related to the limited amount of user data that is in disposition of the user. It is reasonable to expect that the user will require more space for his data.

5. UPT

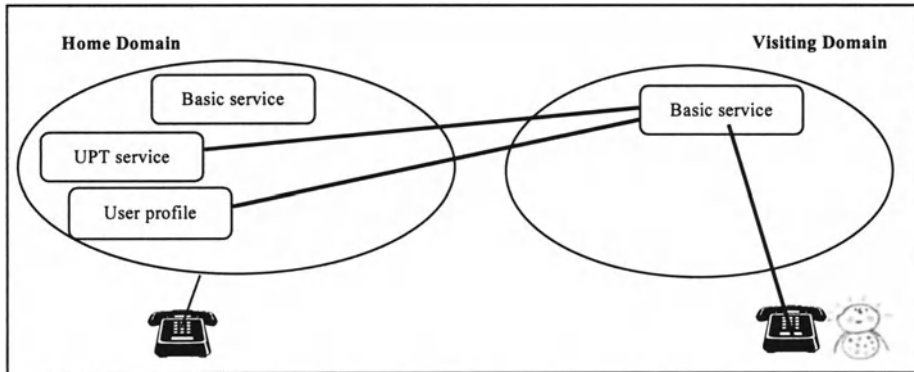


Figure 3. Mobility support in UPT

UPT (Universal Personal Telecommunication) allows the user to make and receive telephone calls from any terminal, fixed or mobile using a unique telephone number called UPT number. UPT can be viewed as a supplementary service built on top of the basic telephony service. Every time the UPT number is dialed, the UPT service is activated and performs the mapping of the UPT number to the number of the telephone that the user is currently used. Before making a call the user registers himself to the UPT service. The UPT service is only installed at the home domain of the user, more specifically in the home Service Control Point (SCP). The user profile is also kept only in the home domain, either in the SCP or the Service Data Point (SDP). UPT does not offer any data storage for the user.

Figure 3 shows a user using a telephone at a visiting domain. To receive phone calls, he registers to the UPT service at home by calling a green number from the visiting domain. Every call request addressed to him arrives to his home UPT service, which deduces the number of his current terminal. Connection can then be established to this terminal. To make a call, the user dials again the green number of his home UPT service and then enters the number to call. Connection is then prolonged from his home domain to the requested terminal

Although the UPT concept is valuable for the user it is today not implemented very efficiently since everything must first come the user home domain before going to his final location. A solution is to deploy the UPT service everywhere but this requires standardization, agreement and collaboration as in GSM.

6. MOBILE IP

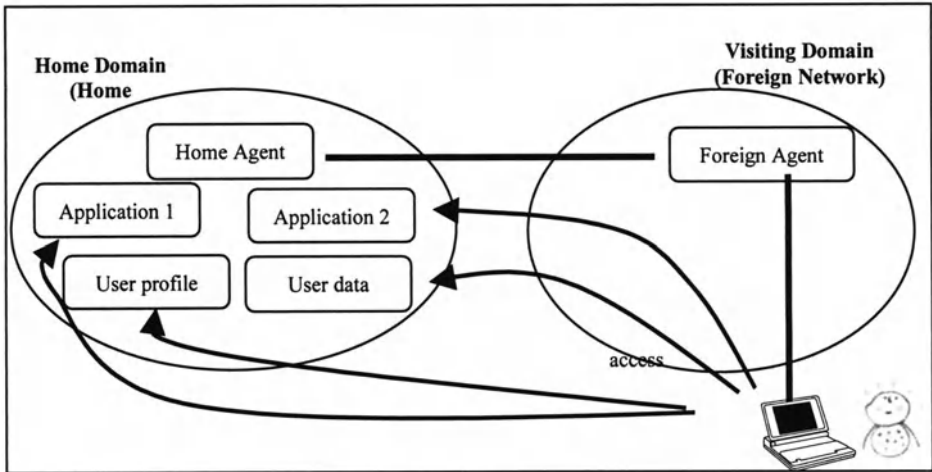


Figure 4. Mobility support in Mobile IP

IP or Internet Protocol is intended for networks of stationary computers or hosts. Each host is permanently assigned to an IP address that contains both the subnet identity and the host identity. There is no support for mobile hosts. The goal of mobile IP is to remedy this problem by allowing hosts to move and still be in communication with other hosts. By communication it is meant that other hosts are still able to send messages or packets to the mobile host and conversely the mobile host is able to send packets to other hosts. Mobile IP allows a mobile host to use two IP addresses. The *home address* is static and denotes the home domain, also called *home network* in mobile IP terminology. The *care-of-address* is dynamic and changes accordingly to the visiting domain, also called *foreign network*, to which the mobile host is currently attached. Whenever the mobile host is not at home, the *home agent* gets all the packets destined for the mobile hosts and forwards them to the foreign agent. The foreign agent arranges to deliver them to the mobile host.

Mobile IP focuses only on providing message communications for the mobile host and does not support the availability of other user applications, user profile or user data. Of course, if everything is stored locally on the mobile host then it will not be a problem. However, it is quite common for a business user to have only a subset of his applications, his data and profile on his laptop while the complete set remains on his corporate home domain. As shown in Figure 4, if the user wants to access applications, profiles or data that reside only at his home domain then he has to change some configuration parameters. This is a cumbersome, time-consuming and quite

boring task which presumes that the user knows the parameters and their new values. The user must also remember the initial values of the parameters in order to be able to restore when coming home.

7. TELNET

In order to access his applications, his data and his profiles the user can also use Telnet, which has existed since the beginning of computer networks. As shown in *Figure 5* Telnet actually establishes a channel from the visiting domain to the home domain and allows the user to access his applications, data and profile located at his home domain. The Telnet protocol provides a standard interface for terminals and terminal-oriented communication. The protocol is normally used over TCP/IP but can also be supported by other protocols. When a Telnet connection is established each side is emulating a "Network Virtual Terminal" (NVT). It is this virtual device that provides the standard interface and hence dictates the look-and-feel. No personalization is possible. When establishing, options requests are sent back and forth as each side tries to negotiate the best possible services. The applications, which can be used from the visiting domain, are restricted by terminal resources, operating system and other restrictions at the application level. As a result, the user can normally access only a subset of his applications. Another disadvantage is the cost and trouble of keeping an open channel over a long period of time.

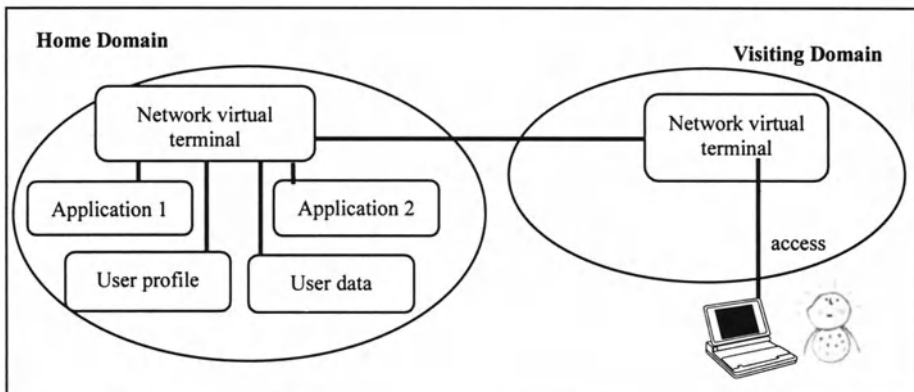


Figure 5. Mobility support with Telnet

8. IMPROVEMENTS BROUGHT BY MOBILE AGENT CONCEPT

From the cases investigated in the previous sections, it is possible to summarize the mechanisms used to implement the mobility support as follows:

1. Enabling access of user applications, profiles and data located at home: This is done by **establishing a channel** from the visiting domain to the home domain as in the case of UPT, Mobile IP and Telnet.
2. **Duplication** of user applications, profiles and data on the visiting domain: This mechanism has again two variants:
 - a. Pre-duplication: This is done in the case of the basic service and some supplementary services in GSM
 - b. Dynamic duplication: This is done in the case of the user profile in GSM. Moving code or logic is quite restricted until the emergence of Java.

All the three mechanisms¹, 2a and 2b have both advantages and disadvantages, and are suitable only for a definite application, a definite situation, a definite visiting environment, a definite usage, etc. It is hence not possible to conclude that one mechanism is generally superior to another. Obviously, a combination of all three mechanisms would be the best solution. Further, such a combination should be dynamic, i.e. could be changed according to the situation. From this observation, the mobile agent concept is very suitable to implement mobility support since it allows all three mechanisms and also all possible combinations of them.

In fact, a strategically mobile agent might be the best solution to the mentioned problem. By strategic we mean that the agent will move according to the user preferences, network load and local resources. At a visiting domain, the user makes a request to his user agent, which is originally at his home domain. The user agent can inquire the visiting domain to obtain resource and capability information available on the visiting domain. The user agent can then travel to the visiting domain bringing along the user's profile. It will then initialize the foreign environment and the user's profile will be available to the user. Depending on different factors such as resources at the visiting domain, quality of the underlying network, type of applications, usage duration, etc. the user agent can decide which applications to move and which to access remotely. The combination used for each case will then be the most optimal according to the criteria defined by the user, the user agent and the visiting domain.

9. AN AGENT-BASED SYSTEM FOR MOBILITY SUPPORT

In this section we present the architecture of an agent-based system, which improves the support of mobility.

The whole system is composed of different sites. A site is governed by an administrator and can be composed of one of several networks (subnets) of nodes (hosts). Every site has a **SA** (Site Agent) assuming the administration function. It has the responsibility of giving access rights to the agents, garbage collect finished agents and make sure that the security model is not violated. It also has the responsibility to retract, i.e. to move a user agent according to the need of the user.

A user has subscription at a site. He is represented by a **UA** (User Agent), which acts on behalf of him and takes care of everything when the user is moving. The UA controls three tasks agents: the **AA** (Application Agent) in charge of the user applications, the **DA** in charge of the user data (document files) and the **PA** (Profile Agent) in charge of the user's profiles.

Figure 6 illustrates the case of a user **X** moving to a Visiting site. At his Home site, user **X** subscribes and has access to applications A_A and A_B . Application A_C is available at his home site but he does not use it. He has also the document files D_{X1} , D_{X2} and D_{X3} . His preferences are registered in the profiles P_{X1} , P_{X2} and P_{X3} . When visiting the Visiting site, user **X** needs all his applications A_A and A_B , the data files D_{X1} and D_{X3} , and the profiles P_{X1} and P_{X2} .

When user **X** logs in at the Visiting site. By his user name, which must be a unique identifier, the visiting site agent SA_V resolves his home site address and communicates with his UA_X . UA_X will perform the authentication of user **X**. If the authentication is successful, the DA_V retracts UA_X causing it to move to the visiting site.

When the UA_X arrives and is restored, it starts communicating with the SA_V , negotiating the access rights. The UA_X will find out what applications are already available at the host and how much memory and computing resources are available. It will then present this information to the user and suggests that the applications that the user normally uses that are not present, the users data files based on some criteria and the users profile are fetched. In our case, only application A_A needs to be fetched since application A_B already exists. Also, the data files D_{X1} and D_{X3} , and profiles P_{X1} and P_{X2} needs to be fetched. Based on the user's answer the UA_X retracts the task agents: AA_X , DA_X and PA_X from user **X**'s home site.

When the task agents receive the retraction call, they will start to collect the files that are necessary to implement the requested services. These files might be spread at many different places at the user's home machine or on

his local network. The agents then encapsulate these files and migrate to the remote host. In our case, AA_x takes care of A_A , DA_x of D_{x1} and D_{x3} , and PA_x of P_{x1} and P_{x2} . When the agents are restored at the remote host they distribute the files so the files will reside at the same directories as the user is accustomed to. If there are existing files with the same name, the agent will give that file a different name, and changes it back later when the user logs off.

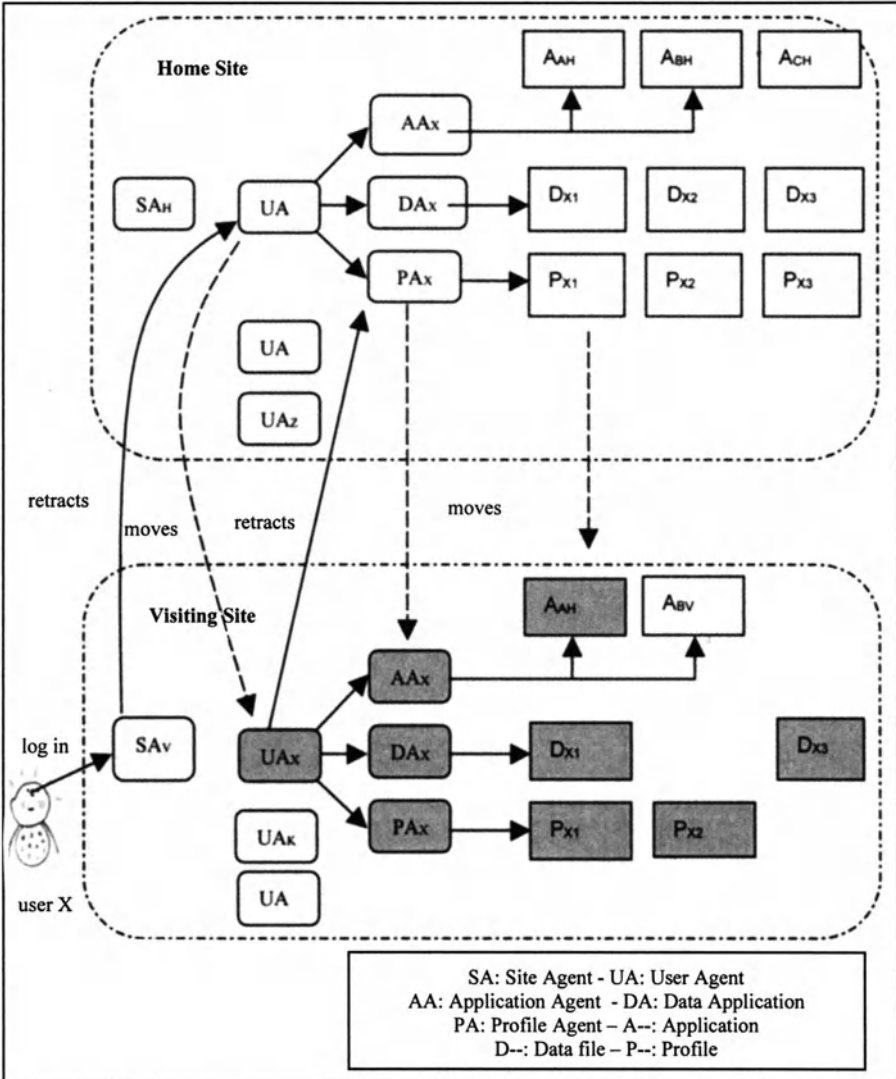


Figure 6. An agent-based system for mobility support

When user X logs off the SA_V will notify the UA_X . The UA_X will then ask the task agents if the files that were brought over at the migration have been changed, if that is the case those files must be returned to the home location before the files are destroyed at the remote site. The UA_X then kills the task agents, before it tells the SA_V that the UA_X has finished and lets the SA_V garbage collect it.

This approach will in our opinion lead to ideal user mobility. We could however achieve an even better result if the agents were strategically mobile. With that we mean that based on the speed of the network, the user's travel patterns and the processing powers at the visiting site the UA could make intelligent choices on how to best satisfy those needs. For instance when the UA migrates from the home site to the visiting site, it could measure the speed of the network connection. Based on this information the UA could indicate how long it would take to move the task agents. In another scenario the UA could ask how long the user is planning to work, based on information about the speed of the network, the UA could estimate the price of keeping an open connection (Telnet) or sending over the task agents. This would be a good solution if the applications were large and the user does not plan to work for a long time.

10. CONCLUSION

In this paper we have shown that the mobility support offered by current mobile systems are far from being ideal for the user and that many improvements can be achieved by using the mobile agent concept. Mobile agents seem to be a promising concept that has the flexibility and the adaptiveness required in a mobile environment. However, as mentioned earlier, in order to be really usable, the mobile agent concept relies on the availability of technologies such as Java chip, Jini, etc. which can equalize the introduced overhead. In addition, there are several issues that also need to be resolved. One crucial issue is security. It is necessary to protect the visiting site from malicious agents and reciprocally, the agent from pirate sites. The user must also be protected so that his private files are not spread on different networks. Another issue also related to security is about firewall. The firewall of a site prevents any intrusion from outside and hence prevents also the visit of mobile agents. Another firewall scheme is needed. Last but not least, since the concept is still new, standards are still lacking despite of the efforts from OMG and FIPA. This can lead to incompatibility between different mobile agent systems.

REFERENCES

- [1] Pål Spilling. The telecom research program telerep at unik. UNIK Aarsrapport 1996, 1996.
<http://www.unik.no/paal/telerep.html>.
- [2] TINA-C. Overall Concepts and Principles of TINA, February 1995.
- [3] ITU-TS. Basic Reference Model of Open Distributed Processing - Part 1 Overview and guide to use the Reference Model. Rec.X901(ISO/IEC 10746-1).
- [4] ITU-TS. Basic Reference Model of Open Distributed Processing - Part 2 Descriptive model. Rec.X901(ISO/IEC 10746-2).
- [5] OMG. The common Object Request Broker: Architecture and specification – Rev 2.2 February 99
- [6] B. Pagurek A. Bieszczad and T. White: Mobile agents for network management, IEEE Communication Surveys, Sept 1998
- [7] (<http://www.objectspace.com/>)
- [8] (<http://www.trl.ibm.co.jp/aglets/>)
- [9] (<http://www.cs.dartmouth.edu/~agent>)
- [10] Crystaliz Inc, General Magic Inc, GMD FOKUS, IBM, TOG: OMG Joint Submission "Mobile Agent System Interoperability Facility", November 1997, available via <ftp://ftp.omg.org/pub/docs/orbos/97-10-05.pdf>
- [11] Asgeir Tomasgard, Jan A Audestad, Shane Dye, Leen Stougie, Maarten H van der Vlerk and Stein Wallace: Modelling Aspects of Distributed Processing in Telecommunication Networks, Annals of Operations Research 82 (1998)161-184.

A MOBILE AGENT-BASED APPROACH FOR THE UMTS/VHE CONCEPT

Peyman Farjami, Carmelita Görg, Frank Bell

Aachen University of Technology, Germany

Institute for Communication Networks

E-mail: {pemi, cg, bell}@comnets.rwth-aachen.de

Key words: Virtual Home Environment (VHE), Universal Mobile Telecommunication System (UMTS), Mobile Agent Technology (MAT), Adaptive Profile Manager (APM), Service Provisioning, Service Roaming, Provider Agent (PA), Terminal Agent (TA), User Interface Agent (UIA)

Abstract: The provision and delivery of personalised services across network and terminal boundaries with the *same look and feel* is currently a challenging issue for service provisioning in future telecommunication networks like UMTS (Universal Mobile Telecommunication System). This paper describes an advanced mobile agent-based approach for realisation of the VHE (Virtual Home Environment) concept considering two main aspects namely *service roaming* and *terminal independence*. The first one implicates the accessibility of personalised services in any network the user roams to, while the latter deals with the adaptation of the service presentation to the corresponding end device (mobile phone, Laptop, PDA,...) according to its capacity and system features as well as the specific user profile. The implementation of some service prototypes are described which were developed in the framework of the ACTS CAMELEON project [11,12].

1. INTRODUCTION

The concept of a Virtual Home Environment (VHE) for UMTS (Universal Mobile Telecommunication System) was introduced into the standardisation process for the provision and delivery of personalised

services across network and terminal boundaries with the *same look and feel*. This concept allows the user to be consistently presented with the same personalised features, user interface customisation and services in whatever network, with whatever terminal (within the capabilities of the terminal), and wherever the user may be located.

The ability to provide such flexible service provisioning has many implications for the network in terms of management, control of services, and signalling. A user request for an unknown service will require some means to locate where that service request can be handled or where information can be found on how to handle the service request itself. This may imply that service control is distributed in the system giving rise to complex network management protocols and stringent security requirements.

Current developments in GSM such as CAMEL (Customised Applications for Mobile networks Enhanced Logic), MexE (Mobile station application Execution Environment) and SAT (SIM Application Toolkit) are expected to meet some requirements of the VHE and they will also be adopted for UMTS. However, there is scope for other techniques to be employed to meet all the requirements of the VHE.

In the last few years agent technology has spread to many areas, including user interface, personal assistance, mobile computing, information retrieval, telecommunication services and service/network management.

This new technology offers a promising solution to cope with the complexity of open environments, since agent-based solutions can:

- reduce the traffic load (via the autonomy and asynchronous operations of the agents)
- enable on-demand provision of customised services (via dynamic agent migration from the provider system to the current user terminal)
- increase the flexibility, reusability and effectiveness of the software-based problem solutions
- offer the potential to distribute service related processing and a mechanism for nodes in different networks to co-operate in order to provide a service to a user

With these advantages agent technology can be used in the following areas of the VHE concept.

- Usage of agents for value added service subscription and provision to the user, such as dynamic migration of applications between the user's mobile terminal and the value added service provider. A mobile software agent transports the software to the current terminal and executes it there.

- Usage of agents for mobile communication service provision to the user, such as dynamic downloading of customised mobile communications service logic into terminals. The idea is to provide customised communication capabilities dynamically to the user and increase the intelligence of the terminal.
- Usage of agents for user and service roaming. A mobile software agent can follow the roaming user, even between different mobile communication systems, to represent the user in the foreign network and provide the user's subscribed services.

However, this technology is relatively new and its suitability for solving telecommunications problems needs to be proven in implementations and through performance analysis. This paper deals with a possible service architecture for personalised service portability in UMTS based on agent technology.

2. AGENT TECHNOLOGY

An *agent* is defined as a piece of software that is able to perform a task autonomously on behalf of a user or application, using its intelligence to access distributed resources. A *place* is an execution environment for agents at a specific location. Two types of agents are distinguished: Provider Agents (PA) and Service Agents (SA). A provider agent is permanently available at a fixed location and offers access to local resources. More than one provider agent can exist at a place. The agent that has been given a certain mission and will be using the services of a provider agent is called the *service agent*. The mission of the service agent is a user-defined abstract set of rules in which the agent should act. A service agent can either be generated at a place locally (possibly out of a pool of ready-programmed usable objects) or can be a visitor that has been created elsewhere and was moved to the place, carrying with it the previous state of execution.

2.1 Voyager as underlying agent platform

The development and prototyping of CAMELEON services are based on the *Voyager* [13] agent platform. The ObjectSpace Voyager platform is a development platform and object request broker for distributed Java-based applications. It allows the usage of a regular message syntax to construct remote objects, messaging between distributed objects and code migration.

The core item of any agent system designed using the Voyager development platform is the host and execution environment called the

Voyager server. Starting a Voyager server is a prerequisite for running Java programs that use the remote functions provided by the Voyager system. The Voyager server is the „place“ that hosts permanent *provider agents* and temporary *service agents*.

In a distributed computing scenario, it is necessary to have a Voyager server installed on every machine that will take part in order to play the role of a „place“ for permanent and temporary agents. A Java Development Kit (JDK) has to be available on any such machine as well. Voyager servers can be invoked manually or from within a Java program. In both cases, a user must have the authority to log into that machine in order to start the server. A server cannot be started remotely. However, once started, the server allows execution of remote software objects from users that have no authority to log into the machine hosting the server.

Voyager servers can act either as a “server” or a “client”. A Voyager server acting as a *client* exerts the function of a „home place“. It allows, e.g., the launching of agents, their dispatching to other places and it also enables them to return. However, acting as a client it cannot receive messages or host software objects other than the ones created there. A Voyager server acting as a *server* can host any remote software object, handle messages to those objects and even act as a code server to distribute class files for remote locations.

The main features of the ObjectSpace VoyagerPro are described in the following:

Using remote objects: A remote object can be used within a local Java program, if a handle to this object exists. In VoyagerPro this handle is called a *proxy*, meaning that it exists locally and the methods of the remote object can be used as if they were local.

Remote construction: It is possible to create a remote instance of any class and obtain a proxy to the newly created object. The proxy implements the same interfaces as the created object and the proxy class is generated dynamically if it doesn't already exist.

Naming service: The VoyagerPro naming service is called *Namespace*. Each object that wants to make its methods available to other objects must bind its URL (Uniform Resource Locator as used for the World Wide Web) to the Namespace. When a program looks up a remote object in this namespace, the URL must be given and, upon finding the object, a proxy to the object is returned. VoyagerPro also supports other naming services like: CORBA, JNDI, RMI registry, etc..

Messaging: VoyagerPro distinguishes between two kinds of messaging: *Remote Messaging* and *Advanced Messaging*. Remote Messaging is defined as method calls made to a proxy, that are forwarded to its object. If the object is in a remote program, the arguments are serialised and later de-

serialised at the destination. Advanced Messaging is a VoyagerPro messaging mechanism for synchronous and asynchronous messaging.

Autonomous mobile agents: With VoyagerPro autonomous mobile agents can be created that move themselves between different hosts and continue to execute upon arrival.

Dynamic class loading: Classes can be dynamically loaded from one or more locations when necessary. This allows to set up class repositories that serve a distributed system with Java applications.

3. INTEGRATION OF MAT INTO UMTS

The introduction of Mobile Agent Technology (MAT) to UMTS assumes some enhancements of the existing network components. As UMTS is an evolutionary concept, which means that it should benefit from the research activities and experiences made within GSM, it is interesting to ask, in how far current network components should be replaced or enhanced to new interfaces. In Phase 2+ of GSM the CAMEL Technology, based on IN concepts, is being introduced. Therefore it is reasonable to keep the existing functionality of CAMEL, but to implement different interfaces corresponding to the MAT.

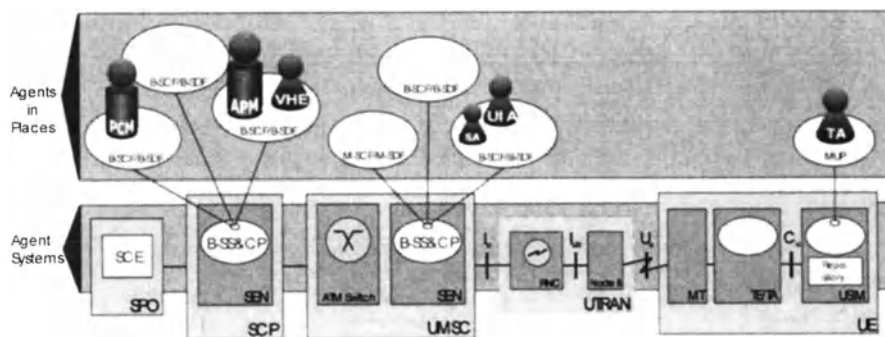


Figure 1. Agents in UMTS

The overall architecture of UMTS integrating the MAT with the corresponding agents is illustrated in figure 1. It contains the network components with MSC (Mobile Switching Center) functionality known from the GSM architecture like mobility management, bearer control, etc.. However, the MSC in UMTS will be more integrated than in GSM. It should manage different access networks, e.g., existing GSM BSSs (Base Station Subsystem) and UMTS RANs (Radio Access Network). It is also foreseen to

integrate an SGSN (Serving GPRS Support Node) node. The transport technology for Signalling and Data over the I_{u} interface is expected to be ATM. The core network will be based on ATM technology as well. These integrated MSCs called UMSCs will be connected to one or more SCPs (Service Control Point), which are able to provide CAMEL functionality.

The user will be equipped with a terminal which contains the required components managing the radio interface MT (Mobile Termination) and indicating the information TE/TA (Terminal Equipment/Adapter). As mentioned before, for the integration of MAT into UMTS the SSPs (Service Switching Point) and SCPs of the UMSCs should be enhanced by additional interfaces. Hereby, the functionality of SCFs (Service Control Function) and SDFs (Service Data Function) are partially merged together for providing a distributed architecture. These combined SCFs and SDFs are called B-SCF/B-SDF, which provide interfaces for broadband communication services. For an agent-based mobility management the so called M-SCF/M-SDF are required, providing an execution environment for an agent migrating to corresponding UMSCs in case of a location update. Additionally, the SENs (Service Execution Node) should provide an execution platform for agent systems called B-SSP/SCPs. The corresponding agent execution environment should also be provided by the USIM or by the TE/TA. The logical conjunction of the distributed agent execution environments (DAE) mentioned above contains the required agents and their service logic residing in different places and communicating through the B-SCF/B-SDF interfaces.

To provide the accessibility of personalised services in any network the user resides in, a so called VHE agent can follow its associated user wherever he roams to. The VHE agent contains the list of a user's subscribed services and its configurations. A Provider Agent (PA), which resides within every serving environment, contains the knowledge of all services provided by this network. So the VHE agent has to connect the provider agent when the user roams to a new serving environment in order to get information on new services provided by the visited network.

As mentioned above the UMSCs have to provide an execution environment (place) which allows the corresponding agents to perform their tasks. This is the assumption for the VHE agent, in order to roam between the different UMSCs.

Additionally, as another strategy the VHE agent may be placed on the user terminal and migrate to the network after a connection establishment in a new network.

In order to choose the best strategy performance analysis is necessary. Within the framework of the CAMELEON project a performance study was carried out which considers the possible options for the placement and

employment of the agents involved in the corresponding scenarios. The performance assessment results are presented in [8,9].

4. AGENT-BASED SERVICE PROVISIONING

For the adaptation of the service presentation to the corresponding end device (mobile phone, Laptop, PDA,...) according to its capacity and system features as well as the specific user profile a so called (APM) Adaptive Profile Manager has been developed and prototyped within the framework of the CAMELEON project. The APM focuses mainly on the terminal independent service provisioning which is an essential aspect of the VHE concept and will offer the following capabilities:

- Flexible service provisioning
- Remote, network and platform independent modification of all user profiles for different services
- Handling of different terminals with different capabilities for service access
- Automatic download of updated services

A terminal agent (TA) on the currently used terminal provides an interface to the user to authenticate himself and request a service (see Figure 2). The terminal agent maintains knowledge of the user terminal profile of the corresponding end device considering its capacity and system features (display type, memory, available bit rate etc.).

On the service provisioning side the APM Provider Agent (APM-PA) manages the user authentication, the profile adaptation and service requests. It is connected to a central register, where all user data are stored. Each service to be accessed by the user should be registered at a APM. This is performed by the communication between the APM-PA and a so called Service Specific Provider Agent (SS-PA). The services could be implemented on different platforms running on different operating systems, only the APM-PA and SS-PA have to communicate and understand each other.

Before responding to a service request the APM-PA has to verify whether the terminal agent on the connected terminal is up-to-date or should be updated. If so, the new version of the terminal agent or sub-agents for specific features can migrate to the terminal and replace the old ones. This automatic version control enables the user to always have the current version of the terminal agent without the need of manual downloading.



Figure 2. Terminal Agent

Once the suitable version of the terminal agent is available on the terminal the user is informed about the currently available services, after a successful authentication process. If there are some new services which the user doesn't yet know he has the possibility to get a demo version of the selected service in order to decide whether he wants to subscribe to it or not.

If the user subscribes to a new service it will be added to the list of subscribed services within the TA (see Figure 2) and the required data for the service subscription are sent from the TA to the APM-PA to be stored in the central register.

Starting a subscribed service, the TA sends the service request to the APM-PA to check whether the user is authorised to access the required service. If so, the APM-PA instructs the so called User Interface Agent (UIA) to transfer the appropriate service specific GUI to the requesting terminal.

The chosen GUI is adapted to the network capabilities and the features of the currently used terminal. The required data concerning the network and terminal capabilities are delivered by the terminal agent when connecting the APM-PA at the beginning of the session.

After the UIA arrives at the terminal the data connection to the APM-PA can be dropped. Here, it can be distinguished between 2 possibilities depending on the realisation of the requested service:

First, the complete service logic can be moved to the terminal; second, the service logic is partially moved to the terminal, in this case a connection (RMI or messaging) between the UIA and the SS-PA is required.

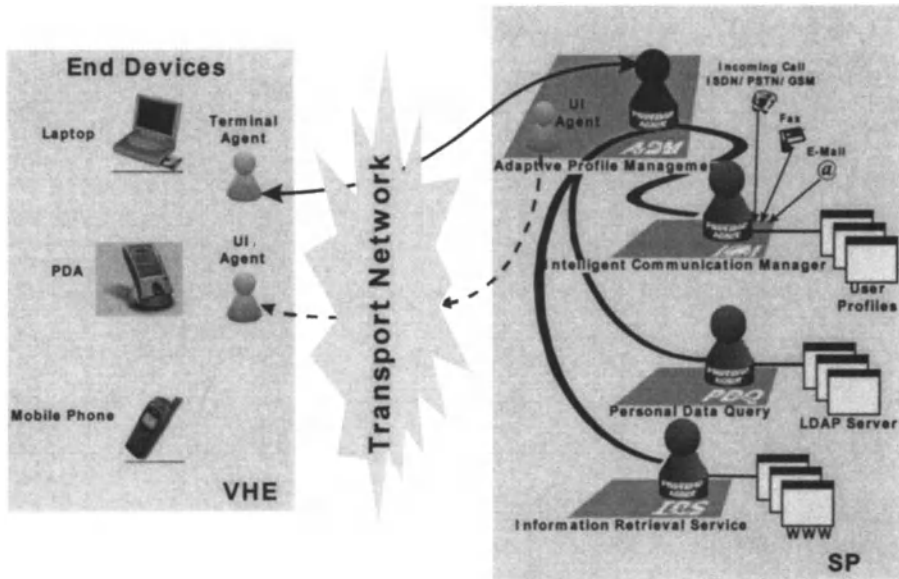


Figure 3. Terminal independence service access

The performance can be enhanced by different methods, e.g., using caching mechanisms to keep the agent code for future service requests and just move the data.

5. SERVICE PROTOTYPES

Currently, the APM provides access to three services namely ICM (Intelligent Communication Manager), PDQ (Personal Data Query), and IRS (Information Retrieval Service) which can be accessed terminal independently and can be presented adaptively to the capabilities of the current terminal and network connection. However, more new services may be added flexibly to the system (see Figure 3).

5.1 Intelligent Communication Manager - ICM

The ICM is the core service of the APM system providing a personal communication environment to the user in order to have a flexible and personalised mechanism to control the reachability for all subscribed communication services.

This service offers to the user the possibility to organise and automatically control any kind of incoming communication requests (voice,

fax, email, SMS, etc.). The management of personal communication is based on the specification of special conditions (caller/sender, time, subject, etc.) in the so called user profiles (filter scripts), that can be configured remotely and adapted to the current requirements of the user.

The ICM's main features are as follows:

- Call (voice, fax, email) recording
- Call rejection
- Caller/application interactions via DTMF-detection or voice menu
- Call forwarding/deflection
- User registration/de-registration
- User notification, e.g., via pager service, SMS, or email
- Conversion of recorded voice- and fax messages for sending them as email
- Forwarding of incoming email to a fax machine

A multi-media mailbox informs the user about the recent personal communication activities performed by the ICM, e.g. in order to view the received faxes or emails and to play the recorded voice messages [4].

5.2 Personal Data Query - PDQ

This service offers to the user the possibility to query data from a personal database by SMS (e.g., address, telephone/fax/mobile phone number, email address, birthdays and others).

The personal data are stored in an LDAP (Lightweight Directory Access Protocol) database which can be accessed by the PDQ service through JNDI (Java Naming and Directory Interface).

This information can either be stored on a local LDAP server or on any LDAP server which is accessible via the Internet.

An incoming SMS contains a personal data query which should be performed by the LDAP server. The results in turn will be sent to the user via SMS [7].

There are several possibilities to send and receive SMS messages:

- Via air interface if the server is connected to a GSM module
- Sending and receiving SMS messages as email
- Using the ISDN gateway of the SMSC to send SMS messages

5.3 Information Retrieval Service - IRS

This service itself is a totally agent-based service offering the user an efficient retrieval service for information, that is distributed over different platforms in heterogeneous networks. An example is the automatic booking of flights or other travel arrangements in the Internet. The user delegates certain functions, e.g., find a flight at a date and time to a certain city, without having to know about details of the task execution. Any information service (e.g., WWW pages, LDAP databases, etc.) with data important for the user can be accessed. The mobile agent with a certain degree of autonomy collects the information and returns with an evaluated result back to the current terminal of the user.

6. IRS SCENARIO

All services (ICM, PDQ and IRS) explained above can be accessed over the APM offering the adaptation of the service presentation according to the end device specific parameters [1,2,7].

The following scenario describes the usage of the Information Retrieval Service through the APM illustrated in Figure 4.

After the user authentication and service request through the terminal agent (TA) on a Laptop the corresponding GUI is transported to the terminal (Laptop) by the user interface agent (UIA). Now the user enters the data he is looking for (e.g. find the best flight offer to London on next Monday). After sending the agent with the query (mission) the connection can be dropped. After a while the IRS service agent will come back to the IRS provider agent to deliver the found information. To present the retrieved result on the terminal the UIA has to move to the terminal. As the terminal is not connected to the network the UIA is set to the *wait* status by the APM-PA and the results are stored. The next time, the user logs into the system via the TA - it could be another terminal like a PDA for example- the UIA is alerted by the APM-PA and migrates with the appropriate GUI accordingly to the currently used terminal to present the retrieved results.

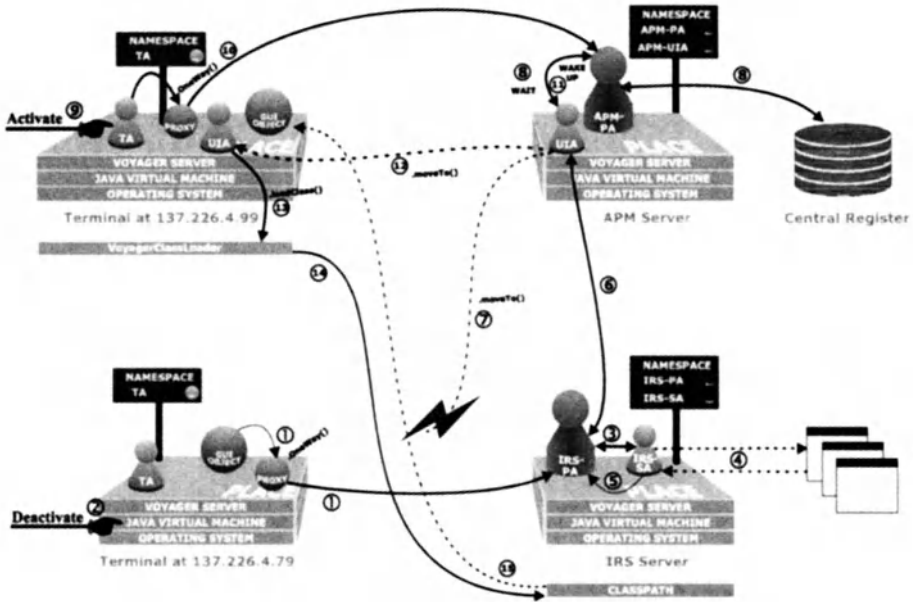


Figure 4. Using the IRS service

All steps through this scenario are described in the following:

1. The UIA sends the entered information to the IRS-PA
2. The user terminates the TA on his current terminal
3. The IRS-PA activates the IRS-Service Agent
4. The IRS-Service Agent moves to the corresponding content provider in order to accomplish its task
5. The found results are passed to the IRS-PA
6. The UIA is instructed to bring the results to the user
7. First, the UIA attempts to move to the terminal from which the user sent his query
8. The UIA is set in the *waiting* mode, as the user is not available at his former terminal any more
9. The user starts the TA on another terminal
10. The user and terminal data are sent to the APM-PA
11. The UIA is alerted by the APM-PA and receives the updated user and terminal data
12. The UIA moves to the current terminal of the user
13. The UIA attempts to load the required classes
14. The classes are loaded from the IRS server
15. The results can be presented according to the capacities of the current terminal

7. CONCLUSION

Pre-UMTS systems have largely standardised the complete set of tele-services, applications and supplementary services. As a consequence, re-engineering is often required to enable new services to be provided and this makes it more difficult for operators to differentiate their services. UMTS will standardise service capabilities and not the services themselves.

The results and experiences in the CAMELEON project have shown that agent technology is a very promising field which can be integrated with other technologies (e.g., MExE, SAT, CAMEL/IN) in order to allow flexible service provisioning, execution and access for future telecommunication networks and services. These agents can use the same mechanisms for communication and thus allow a very flexible service architecture. This will lead to more reliable and performant services. Mainly security and further performance aspects should be investigated for a widespread and efficient employment of these technologies in future networks.

ABBREVIATIONS

APM	Adaptive Profile Manager
APM-PA	Adaptive Profile Manager-Provider Agent
AWT	Abstract Window Toolkit
BSS	Base Station Subsystem
CAMEL	Customised Applications for Mobile networks Enhanced Logic
CAMELEON	Communication Agents for Mobility Enhancements in a Logical Environment of Open Networks
DAE	Distributed Agent Environment
GSM	Global System for Mobile communications
GUI	Graphical User Interface
ICM	Intelligent Communication Manager
IN	Intelligent Network
IRS	Information Retrieval Service
JDK	Java Development Toolkit
JNDI	Java Naming and Directory Interface
LDAP	Lightweight Directory Access Protocol
MAT	Mobile Agent Technology
MExE	Mobile station application Execution Environment
MSC	Mobile Switching Center
MT	Mobile Termination
PDA	Personal Digital Assistant
PDQ	Personal Data Query
RAN	Radio Access Network
SA	Service Agent
SAT	SIM Application Toolkit
SCF	Service Control Function

SCP	S ervice C ontrol P oint
SDF	S ervice D ata F unction
SDP	S ervice D ata P oint
SEN	S erving E xecution N ode
SGSN	S erving G PRS S upport N ode
SMS	S hort M essage S ervice
SMSC	S hort M essage S ervice C enter
SS-PA	S ervice S pecific- P rovider A gent
TA	T erminal A gent
TE/TA	T erminal E quipment/ T erminal A dapter
UIA	U ser I nterface A gent
UMTS	U niversal M obile T elecommunication S ystem
VHE	V irtual H ome E nvironment

REFERENCES

1. F. Bell, Concept and Implementation of an Agent Based and End Device Independent Access Mechanism for the Telecommunication User, Master Thesis, Communication Networks, University of Technology Aachen, March 1999.
2. M. A. Döll, Requirement Analysis, Concept and Realisation of an Agent based Platform-Independent Information Retrieval System, Master Thesis, Communication Networks, University of Technology Aachen, October 1998.
3. ETSI TS 22.70 Universal Mobile Telecommunication System (UMTS), Service Aspects: Virtual Home Environment (VHE). UMTS Draft Version 3.0.0, 1998.
4. P. Farjami, S. Kluwe: Java-Based Personal Communication Management. In ITG-Fachbericht 153 Internet - frischer Wind in der Telekommunikation, pp.193-200, ITG-Fachtagung, Stuttgart, October 1998, VDE-Verlag
5. FIPA. Foundation for Intelligent Physical Agents, ACL Agent Communication Language. Specification, part 2., 1997.
6. J. Hartmann, C. Görg, P. Farjami: Agent Technology for the UMTS VHE Concept. In Proceedings of the First ACM International Workshop on Wireless Mobile Multimedia in Conjunction with ACM/IEEE MobiCom'98, pp. 203-208, University of North Texas, Dallas, USA, October 1998, ACM
7. L. Holthausen, Conception and Implementation of an Information Service Using a Personal Information Server Based on the Short Message Service (SMS) of the Mobile Networks, Master Thesis, Communication Networks, University of Technology Aachen, October 1998.
8. R. Lemle, Concept and Evaluation of Agent-based Architectures for the Virtual Home Environment (VHE) of Future Mobile Telecommunication Networks (UMTS), Master Thesis, Communication Networks, University of Technology Aachen, August 1999.
9. H. Long, Performance Analysis of Mobile Agents for Future Telecommunication Services, Master Thesis, Communication Networks, University of Technology Aachen, February 99.
10. FIPA. Foundation for Intelligent Physical Agents <http://www.cselt.stet.it/fipa/>
11. CAMELEON project <http://www.comnets.rwth-aachen.de/~cameleon>
12. CLIMATE ACTS Cluster for Intelligent Mobile Agents in Telecommunication Environments <http://www.fokus.gmd.de/cc/ima/climate/>
13. Voyager agent platform <http://www.objectspace.com>

A NOVEL EFFICIENT AND ROBUST EXPLICIT RATE CONGESTION AVOIDANCE ALGORITHM FOR ABR SERVICE IN ATM NETWORKS

Tanun Jaruvitayakovit, Naris Rangsinoppamas,
Boonchoung Tansuthepverawongse and Prasit Prapinmongkolkarn
*Telecommunications System Research Laboratory, Department of Electrical Engineering,
Faculty of Engineering, Chulalongkorn University, Phaya Thai Road, Bangkok 10330, Thailand*

Key words: ABR service, Explicit Rate (ER) calculation, Max-min fairness

Abstract: This paper investigates the problems in explicit rate congestion avoidance algorithms for ABR service in ATM networks. It is found that the performance of the existing algorithms [2, 3, 4] depend on network conditions which vary dramatically in the real condition such as number of traversing sessions, traffic characteristics and propagation delay. In addition, this paper proposes a novel efficient and robust ABR congestion avoidance algorithm called “Fast Rate Allocation Congestion Avoidance (FRACA)”. Some simulation results are presented to illustrate performance in terms of computation accuracy and switch buffer occupancy of the proposed algorithm. Moreover, the simulation results are compared with Explicit Rate Indication Congestion Avoidance (ERICA+) and Enhanced Fast Max-Min Rate Allocation (E-FMMRA) which are the well-known explicit rate switch algorithms and also compatible with ATM Forum Traffic Management Specification 4.0. From comparison, the proposed algorithm works better than both algorithms for all configuration scenarios.

1. INTRODUCTION

Available Bit Rate (ABR) service in ATM networks was designed in standard body of Traffic Management [1] for supporting applications that tolerate to delay but sensitive to cell loss. ABR service category uses close-

loop feedback for flow control. Resource Management (RM) cell is sent to network by the sources after sending Nrm-1 user data cells to the network. The destination turns around and sends the RM cell in backward direction which is known as backward RM cell. The intermediate switches compute the proper rate and convey it in the Explicit Rate (ER) field in RM cell. After receiving backward RM cell, all sources adjust their traffic rate in accordance with the feedback information in backward RM cell.

The main goals for switch algorithm design are the share of free bandwidth (left over from the higher priority traffic such as CBR and VBR services) to all sources in a fair manner, maximize network utilization with controllable switch queue length. Another important topic for the ABR service is that the above goals should be achieved regardless of network conditions such as number of traversing sessions, traffic characteristics and system propagation delay. The contribution of this paper is to develop a novel, effective and robust algorithm which can achieve the goals regardless of network conditions unlike the previously proposed algorithms [2, 3, 4] which parameters setting mainly depend on network conditions.

This paper is organized as follows. Section II describes and discusses the problems associated with existing algorithms (ERICA+ and E-FMMRA). Section III describes the proposed algorithm and points out the advantages of the proposed algorithm. In section IV, we used simulation results to illustrate performance of the proposed algorithm compared with ERICA+ and E-FMMRA algorithm. Finally, the conclusion is given in section V.

2. MOTIVATION AND RELATED WORK

ERICA+ [4] and E-FMMRA [2, 3] are the well-known explicit rate ABR congestion control algorithms in ATM networks. Both algorithms are exact fair rate calculation algorithms [7].

Explicit Rate Indication Congestion Avoidance (ERICA+)

The algorithm calculates fair share rate which can be defined as

$$\text{FairShare} = \frac{\text{ABR Target Rate}}{\text{Number of active connections}} \quad (1)$$

where ABR target rate is the multiplication result of the left over bandwidth (from higher priority service class) and hyperbolic queue control function $[f(Q)]$ which defined as

$$f(Q) = \begin{cases} \frac{b \times Q_0}{(b-1) \times Q + Q_0} & \text{for } 0 \leq Q \leq Q_0 \\ \max\left(QDLF, \frac{a \times Q_0}{(a-1) \times Q + Q_0}\right) & \text{for } Q > Q_0 \end{cases} \quad (2)$$

where a and b are fixed parameters set to 1.15 and 1.0, respectively [4]. T_0 , which is converted into the target queue length (Q_0), specifies the target queuing delay. Queue Drain Limit Factor (QDLF) is the parameter to limit the rate of queue drain and set to 0.5. For queuing delays smaller than T_0 , the hyperbola is controlled by parameter b (called b -hyperbola). On the other hand, a -hyperbola determines how much drain capacity is used for draining out the queues built up. More drain capacity is allocated when the queue lengths are larger, up to a maximum of $(1 - QDLF)$.

The algorithm computes load factor (Z) based on ABR input rate and ABR target rate. Utilizing the load factor Z , the term VCShare is calculated as

$$VCShare = \frac{CCR}{Z} \quad (3)$$

where CCR (Current Cell Rate) is derived from the received forward RM cell to ensure that the most current information is used to provide fast feedback. In order to guarantee max-min fairness, the terms MaxAllocCurrent and MaxAllocPrevious are introduced. MaxAllocCurrent is the maximum value of every computed ER for all connections during current averaging interval and also MaxAllocPrevious which is the maximum value of every computed ER for all connections during previous averaging interval. Upon reception of a backward RM cell, the ER field is marked down as follows:

$$\begin{aligned} &\text{if } (Z > 1+\delta) \quad \dots \text{ where } \delta \text{ is a small value, typically set to } 0.1 \\ &\quad \text{ER in BRM cell} = \min(\text{ER in BRM cell}, \max(\text{fairShare}, VCShare)) \\ &\text{else } \text{ER in BRM cell} = \min(\text{ER in BRM cell}, \\ &\quad \max(\text{fairShare}, VCShare, \text{MaxAllocPrevious})) \end{aligned} \quad (4)$$

The performance of the ERICA+ algorithm depends significantly upon the way measurements are done and its parameters chosen [4]. ERICA+ algorithm suffers from achieving convergence in case of a large number of connections [6] as the effect from receiving all BRM cells during different averaging interval. In complex configuration, for example Generic Fairness Configuration 2 (GFC-2) [5], the convergence time is very long. In addition, at steady state the queue level cannot be conducted to the targeted value but it oscillates around the targeted value instead because the working region of the algorithm is in overload condition ($1 < Z < 1+\delta$). The queue control functions (step, linear, hyperbolic) mainly effect performance of the

algorithm [8]. Although with hyperbolic queue control function, the algorithm mostly works well but the computation complexity is very high compared to others.

Enhanced Fast Max-Min Rate Allocation (E-FMMRA)

The algorithm computes the rate it can support which is known as advertised rate (γ). If a connection cannot use the advertised rate, it is marked as bottlenecked elsewhere and its bottleneck bandwidth is recorded. This implies that there is additional bandwidth available that can be shared by other connections. The ER field in the received RM cell is read and marked in both direction to speed up the rate allocation process. The allocated rate to all non-bottleneck sessions is recorded as the maximum value of ER, denoted as $ER_{\max}$. The algorithm computes ER_{adjust} according to (5) every time the switch receives FRM and BRM cell.

$$ER_{\text{adjust}} = \frac{ER_{\max}}{\text{Load factor}} \quad (5)$$

Where Load factor is the ratio between ABR input rate and ABR target rate. The load factor reflects how well the ABR bandwidth is utilized. If switch queue length is lower than the high threshold (DQT), the algorithm updates $ER_{\max}$ according to (6) when the switch receives BRM cell.

$$ER_{\max} = (1 - \alpha) \times ER_{\max} + \alpha \times \max(ER, ER_{\text{adjust}}) \quad (6)$$

α is an averaging factor and set at 1/8. For the heavy congestion condition (queue level is greater than the high threshold (DQT)), $ER_{\max}$ is set to the maximum value between advertised rate and ER_{adjust} , that is

$$ER_{\max} = \max(\gamma, ER_{\text{adjust}}) \quad (7)$$

This means that non-bottlenecked sessions are not given any extra bandwidth (if available) in order to drain the queue to the targeted region. The algorithm updates the ER filed in a RM cell both in forward and backward direction for fast transient response [3]. The ER field in forward direction is updated according to

$$ER \text{ in FRM cell} = \min(ER \text{ in FRM cell}, \max(\gamma, (1 - \beta) \times ER_{\text{adjust}})) \quad (8)$$

where β is a single bit value indicating the session is bottleneck elsewhere. For backward direction, the allocated rate is updated as one of follows

$$ER \text{ in BRM cell} = \min(ER \text{ in BRM cell}, \max(\gamma, (1 - \beta) \times ER_{\text{adjust}})) \quad (9)$$

$$\text{or } ER \text{ in BRM cell} = \min(ER \text{ in BRM cell}, \max(\gamma, (1 - \beta) \times ER_{\max})) \quad (10)$$

$$\text{or } ER \text{ in BRM cell} = \min(ER \text{ in BRM cell}, \gamma) \quad (11)$$

The backward rate allocation (9-11) is employed upon the level of network congestion (switch queue length and load factor). If the switch is

not congest, queue level is lower than the set low threshold (QT), or switch is in moderate congestion condition, queue level locates between QT and DQT, and load factor is greater than one, the switch allocates ER according to (9). If switch is in moderate congestion condition and load factor is lower than or equal to unity, the switch computes ER regards (10). Finally if switch is in heavy congestion condition, the allocated rate is computed regards (11).

Actually sources traffic characteristic mainly effects performance of the algorithm. If all sources are persistent, the algorithm always converge to the max-min fairness with very fast transient response. In the case of source-bottleneck configuration, some sources send their traffic at the rate below the proper rate, the algorithm diverges from the targeted working point as the effect from setting too short averaging interval (as recommend in [3]) and updating forward and backward RM cell every time the switches receive. Moreover, we also found other two drawbacks of the algorithm. First, the algorithm suffers from achieving convergence in multi-link rate configuration as the effect of marking both forward and backward RM cell. Second, the algorithm cannot actually be forced to drain the switch queue level to the targeted region when all sessions are non-bottleneck sessions.

From robustness point of view, the performance of the congestion control algorithm should not depend on network condition, the more realistic case. In the other word, the algorithm should converge to the targeted working point with fast transient response regardless of network condition, for example number of connections, sources traffic characteristic and propagation delay.

3. THE FAST RATE ALLOCATION CONGESTION AVOIDANCE (FRACA) ALGORITHM

The main goals for developing the novel congestion control algorithm is convergence according to max-min fairness criterion with very fast transient response, 100% link utilization and controllable switch queue length with the simplest queue control function regardless of network conditions. The algorithm works only in backward direction and does not use CCR field in RM cell to compute the feedback rate. The connections which cannot use the advertised rate ($A_{\max}$) are marked as non-bottlenecked connection (at this switch) and their bandwidth is recorded. The rest bandwidth is equally shared among bottlenecked connections (at this switch). Marking only backward RM cell means we let only upstream switches to know status of downstream switches. However load factor is employed in order to increase network utilization in case there is some connections send their traffic at the rate below the proper rate and upstream bottleneck scenario. Moreover,

system parameters are updated once per connection per averaging interval which set long enough in order to accurately estimate traffic characteristic while maintaining fast transient response. From our study, setting averaging interval to 1500 cells works well for all scenarios (LAN, WAN and satellite ATM networks).

Basically, count-based setting of average interval (timer will be expired if the amount of ABR data cells reach the set threshold) is better than time-based setting of average interval especially if works with background VBR traffic. [6] illustrated that congestion control algorithm which uses per averaging computation and working region located in overload condition (ex. ERICA+) cannot work well with count-based averaging interval. So, the working region of the proposed algorithm is set at load factor equals unity (input rate equal served rate) and uses common weighted averaging for computation the feedback ER according to (6).

Non-bottlenecked sessions receive advertised rate as allocated ER. That is

$$ER \text{ in BRM cell} = \min (ER \text{ in BRM cell}, A_{\max}) \quad (12)$$

Otherwise, the bottlenecked sessions get $ER_{\max}$ as feedback ER.

$$ER \text{ in BRM cell} = \min (ER \text{ in BRM cell}, ER_{\max} / Z) \quad (13)$$

Moreover, we use switch queue level as the additional parameter for computation the feedback ER. The proposed algorithm works well with step queue control function, the simplest queue control function [8]. The goal of queue control is keeping stable non-zero queue level at steady state. There are two important reasons for keeping stable non-zero queue level. First, queue control function is used to compensate for measurement and feedback errors caused by the system. Second, for configuration which has background VBR traffic, it is more reasonable to keep queue level at small level (non zero) because these queue will drain simultaneously in the case VBR traffic is in off phase (no need to wait for BRM cells that will take time to inform the sources) imply 100% link utilization. Our algorithm uses 2 levels which are low threshold (QT) and high threshold (DQT) for controlling queuing delay. The main objective is to keep switch queue length between QT (500 cells) and DQT (2000 cells). The algorithm computes $adjusted\_ER$ which used to control queue level as follows

$$adjusted\_ER = \frac{(f(Q)-1) \times ABR \text{ Capacity}}{N - N_{bottlenecked}} \quad (14)$$

where $f(Q)$ is the step queue control function shown in Table 1, N is number of established connections and $N_{bottlenecked}$ is number of bottlenecked connections.

Table 1. Step queue control function

Switch queue length (cells)	$f(Q)$
$0 \leq Q \leq 500$	1.02
$500 < Q \leq 2000$	1.00
$2000 < Q \leq 3000$	0.98
$3000 < Q \leq 4000$	0.95
$4000 < Q \leq 5000$	0.90
$5000 < Q \leq 6000$	0.80
$6000 < Q$	0.60

For the bottlenecked connections

$$\text{ER in BRM cell} = \text{ER in BRM cell} + \text{adjusted\_ER} \quad (15)$$

The queue control function is also employed to compute load factor (Z) as follows

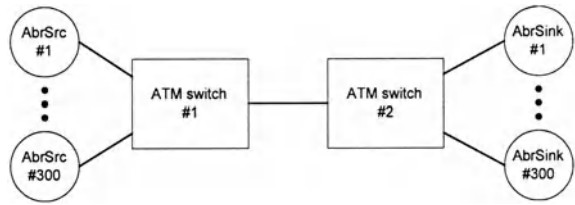
$$Z = \frac{\text{ABR Input Rate}}{\text{ABR Capacity} \times f(Q)} \quad (16)$$

4. SIMULATIONS AND DISCUSSIONS

We use 3 network configuration scenarios to illustrate performance of the proposed algorithm compared with ERICA+ and E-FMMRA algorithm. The first configuration “300 source configuration” is employed to present the problem regarding divergence of sources Allowed Cell Rate (ACR) in ERICA+ algorithm leading to growing unbound of switch queue length although the algorithm employs queue control function. The second configuration “Parking-lot + Link bottleneck + Source bottleneck Configuration” is done to show the problems associated with divergence of allocated rates by E-FMMRA algorithm in multi-link rate and parking-lot configuration and also long convergence time in ERICA+ algorithm. The third configuration scenario “Generic Fairness Configuration 2 (GFC-2)” is simulated to show the problems regarding the divergence of allocated rate in E-FMMRA algorithm, convergence time and buffer occupancy in ERICA+ algorithm.

Simulation results

1. 300 source configuration



Parameters setting:

- All sources are persistent sources, PCR = Link capacity = 149.76 Mbps., MCR = 0.0 Mbps., ICR = 0.7 Mbps.
- All link propagation delay = 5 msec.

Figure 1. 300 source configuration

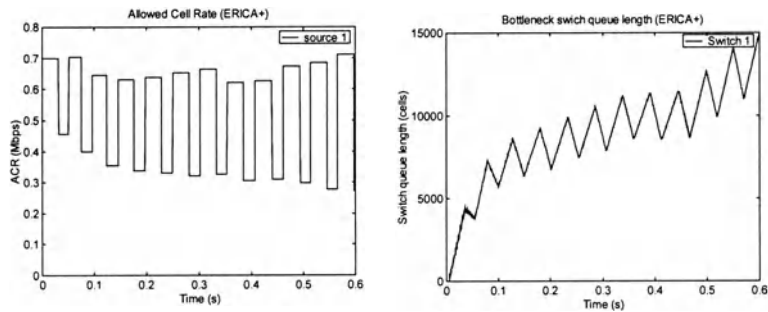


Figure 2. Results for 300 source configuration (ERICA+)

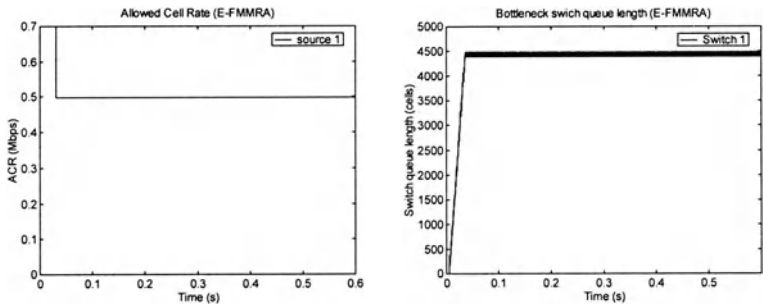


Figure 3. Results for 300 source configuration (E-FMMRA)

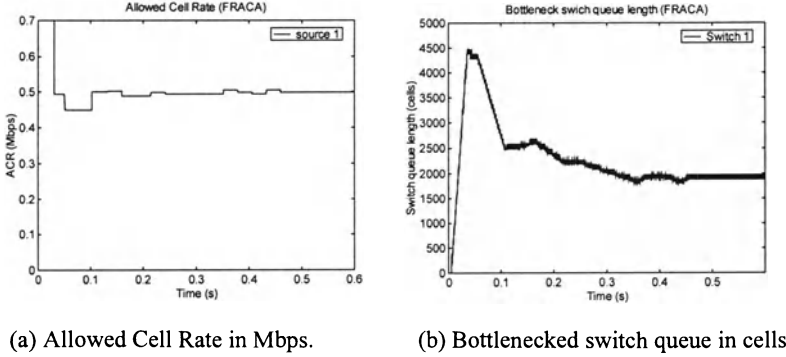
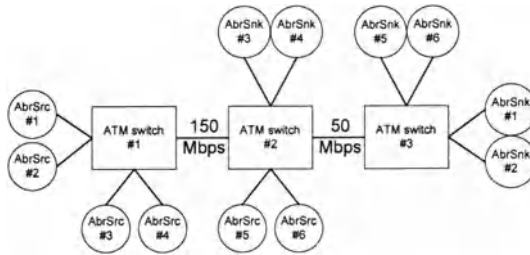


Figure 4. Results for 300 source configuration (FRACA)

This configuration is presented in order to illustrate effects of a large number of VCs to the congestion control algorithms. The expected rate for all connections is $149.76/300 \approx 0.5$ Mbps. From simulation results, the ERICA+ faces the severe problem regarding the divergence of sources ACR as the effect from receiving all sources BRM cells during different averaging interval and the algorithm is per average interval computing [6] leading to growing unbound of switch queue length. For E-FMMRA algorithm, although the allocated rates converge to the expected value but the queue level at the bottleneck switch cannot be conducted to the desired region (QT and DQT is set to 50 and 1000 cells, respectively). For FRACA algorithm, the allocated rates converge very fast to the expected value with controllable switch queue length.

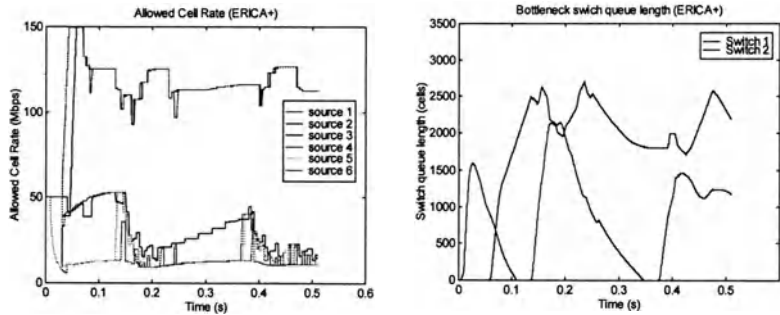
2. Parking-lot + link bottleneck + source bottleneck configuration



Parameters setting:

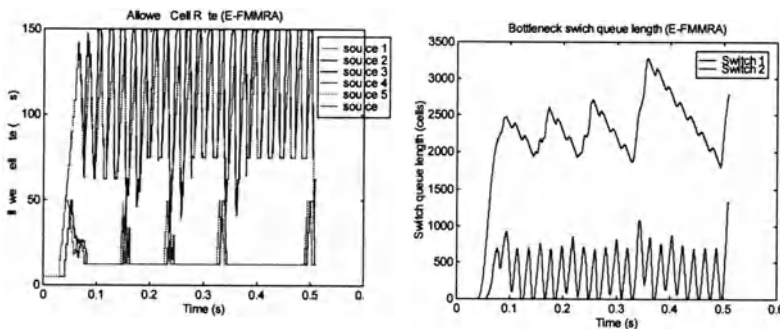
- Sources 1, 3 and 5 are persistent sources, PCR = 150.0 Mbps., MCR = 0.0 Mbps., ICR = 5.0 Mbps., Sources 2, 4 and 6 are bottlenecked sources (send data at rate 10 Mbps. all the time)
- All link propagation delay = 5 msec.

Figure 5. Parking-lot + link bottleneck + source bottleneck configuration



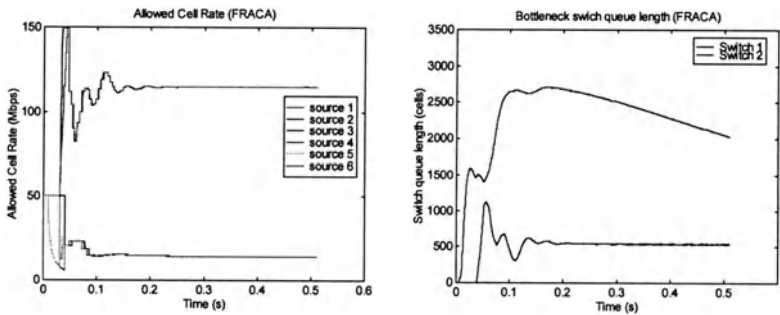
(a) Allowed Cell Rate in Mbps. (b) Bottlenecked switch queue in cells

Figure 6. Results for Parking-lot + link bottleneck + source bottleneck configuration (ERICA+)



(a) Allowed Cell Rate in Mbps. (b) Bottlenecked switch queue in cells

Figure 7. Results for Parking-lot + link bottleneck + source bottleneck configuration (E-FMMRA)



(a) Allowed Cell Rate in Mbps. (b) Bottlenecked switch queue in cells

Figure 8. Results for Parking-lot + link bottleneck + source bottleneck configuration (FRACA)

E-FMMRA faces the severe problems regarding divergence of sources ACR in source bottleneck configuration leading to oscillation of switch queue level in switch#1. ERICA+ performs worse with source bottleneck configuration. The reason is that ERICA+ algorithm uses CCR field in forward RM cell to compute the feedback ER. But for source bottleneck scenario the CCR field does not reflect the actual transmit rate. The following are the expected rates according to max-min fairness criterion.

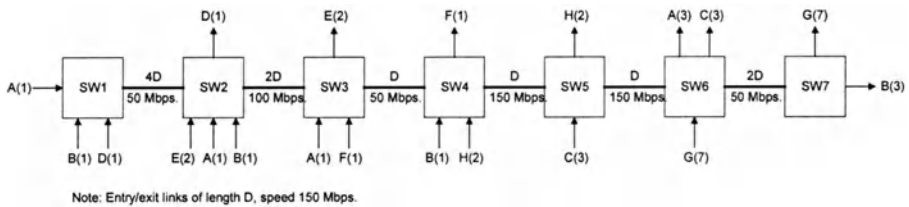
Source#1, #2, #5, #6 = 15.0 Mbps.

Source#3, #4 = 115.0 Mbps.

Recall that source#2, 4 and 6 send data at 10 Mbps. all the time but their ACR field (in FRM cell) are 15.0, 115.0 and 15.0 Mbps, respectively. In the scenario, the ACR field does not reflect the actual transmit rate of these sources.

From simulation results, ERICA+ computes wrong ER especially Source#1, #2, #5 and #6. Although source#3 and #4 ACR converge to the targeted value but the convergence time is very long. Moreover, the utilization of link 2 is not 100 percent all the time (correspond to queue level in switch 2). With the proposed algorithm, all sources ACR converge to the expected rate very fast. The queue level is actually conducted to the targeted region results in full link utilization all the time.

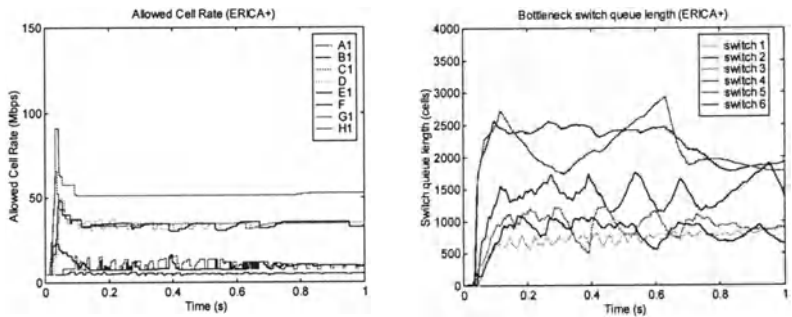
3. Generic Fairness Configuration – 2 (GFC-2)



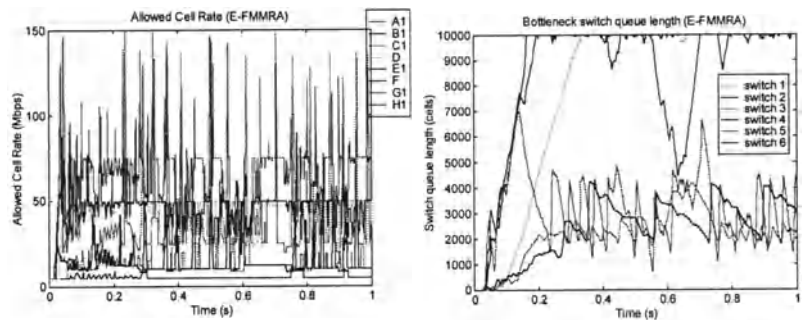
Parameters setting:

- All sources are persistent: PCR = 150.0 Mbps., MCR = 0.0 Mbps., ICR = 5.0 Mbps.
- Link propagation delay (D) = 2.5 msec.

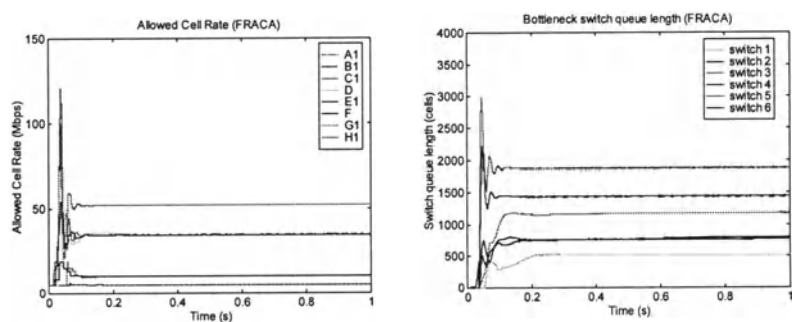
Figure 9. Generic Fairness Configuration –2 (GFC-2)



(a) Allowed Cell Rate in Mbps. (b) Bottlenecked switch queue in cells
Figure 10. Results for GFC-2 configuration (ERICA+)



(a) Allowed Cell Rate in Mbps. (b) Bottlenecked switch queue in cells
Figure 11. Results for GFC-2 configuration (E-FMMRA)



(a) Allowed Cell Rate in Mbps. (b) Bottlenecked switch queue in cells
Figure 12. Results for GFC-2 configuration (FRACA)

GFC-2 is one of the contribution scenario in ATM Forum [5] for testing fairness and robustness of congestion avoidance algorithm. The following are the expected rate allocation according to max-min fairness criterion.

A VCs each get 1/4 of 40 Mbps.	= 10.0 Mbps.
B VCs each get 1/10 of 50 Mbps.	= 5.0 Mbps.
C VCs each get 1/3 of 105 Mbps.	= 35.0 Mbps.
D VC gets	= 35.0 Mbps.
E VCs each get 1/2 of 70 Mbps.	= 35.0 Mbps.
F VC gets	= 10.0 Mbps.
G VCs each get 1/10 of 50 Mbps.	= 5.0 Mbps.
H VCs each get 1/2 of 105 Mbps.	= 52.5 Mbps.

E-FMMRA algorithm diverges from the expected working region as the effect of multi-link rate and parking-lot configuration. The convergence time of ERICA+ is very long although the algorithm converges to the expected working region. Moreover, the queue level is cannot actually conducted to the desired value (5 msec. $\approx$ 1766 cells in 149.76 Mbps. link) but it oscillates around the desired value instead as the effects from overload condition of the algorithm. For the proposed algorithm, sources ACR converges to the expected rate very fast with constant controllable queue level.

5. CONCLUSION

This paper presents a novel efficient and robust explicit rate congestion control algorithm for ABR service in ATM networks. The main goals for designing the proposed algorithm are convergence of sources ACR to the targeted working point according to max-min fairness with very fast transient response, 100% link utilization and controllable switch queue length with the simplest queue control function regardless of network condition and also should not affected by slow start phase of TCP application. We used 3 network configurations to illustrate performance of the proposed algorithm and also compared with ERICA+ and E-FMMRA, the well-known exact fair rate calculation algorithm. From comparison, the proposed algorithm works better than ERICA+ and E-FMMRA algorithm for all cases. Hence, our proposed algorithm should be appropriate exact fair rate calculation algorithm to implement in ATM switches working in real situation which network conditions vary simultaneously all the time.

6. ACKNOWLEDGEMENT

The authors are very thankful to Dr.Pichani Bodharamik, Dr.Somchai Jitapunkul and Dr.Narong Yoothanom for their encouragement and providing facilities for this research. Also heartfelt thanks are given to

Telecom Consortium Scholarship for giving part of financial support for research to the first author.

REFERENCE

1. "The ATM Forum Traffic Management Specification version 4.0", April 1996.
2. Arulambalam, and N. Ansari, "An Intelligent Explicit Rate Control Algorithm for ABR Service in ATM networks", IEEE International Conference on Communications, 1997.
3. Chiussi, Arulambalam, Ye Xia, Xiaoqiang Chen, "Explicit Rate ABR Scheme Using Traffic Load as Congestion Indicator", Proceeding of Sixth International Conference on Computer Communications and Networks, 1997.
4. Shiv Kalyanaraman, Raj Jain, Sonia Fahmy, Rohit Goyal and Bobby Vandalore, "The ERICA Switch Algorithm for ABR traffic Management in ATM Networks", Submitted to IEEE/ACM Transaction on Networking, November 1997.
5. Robert J. Simcoe, "Test configurations for fairness and other tests", ATM Forum/94-0557, July 1994.
6. Tanun Jaruvitayakovit, Naris Rangsinoppamas and Prasit Prapinmongkolkarn "Improvement of ERICA+ Scheme using Adaptive Average Interval Algorithm", Applied Telecommunication Symposium'99, April 1999, San Diego, USA.
7. Arulambalam, X. Chen, and N. Ansari, "Allocating Fair Rates for Available Bit Rate Service in ATM networks", IEEE communication magazine, 92-100, November 1996.
8. Bobby Vandalore, Raj Jain, Rohit Goyal, Sonia Fahmy, "Design and Analysis of Queue Control Function for Switch Scheme", ATM Forum/97-1087, December 1997.
9. Raj Jain, "Congestion control and traffic management in ATM networks: Recent advances and a survey", Computer Networks and ISDN Systems, 1995.
10. R. Onurval, "Asynchronous Transfer Mode Networks: Performance Issues", Artech House, 1994.

A DISTRIBUTED ROUTING ALGORITHM IN CLOS NETWORK OF VARIABLE BIT RATE TDM SWITCH

Hiroaki Morino, Hitoshi Aida, Tadao Saito

Department of Information and Communication Engineering

Faculty of Engineering, University of Tokyo

7-3-1 Hongo Bunkyo-ku Tokyo 113-8656 Japan

Tel: +81-3-3814-4251 ext. 6762

Key words: Variable bit rate, TDM switch, Video traffic, Clos network

Abstract: In this paper, a new routing method is proposed in Clos network of variable bit rate TDM switch. This method is dynamic routing method, and it has following two features; (1) In a TDM frame period, all timeslots applied to the switch can reach their destination ports without loss due to internal blocking. (2) Buffer for queuing is not required in switches of the first stage. Performance of the method is evaluated by software simulation and it is indicated that it can be controlled by commercially available microprocessor in real-time when size of Clos network is within 16×16 .

1. INTRODUCTION

In recent years, there has been an increasing demand for video communication, including video conferencing and video-on-demand system. With advancement of real-time video encoding technology, cost effective variable-bit-rate (VBR) transmission will have widespread use in the future.

As a suitable architecture for variable bit rate video switching system in the future, we have already proposed variable bit rate TDM video switch (VTDM video switch) architecture [1]. The fundamental characteristics of the switch is that end-to-end delay jitter of timeslot is constant regardless of

offered load. Moreover, large timeslot size and intelligent priority control enables both of efficient transmission and guarantee of traffic quality for variable-bit-rate video traffic.

Today, bit rate of encoded video of standard TV-quality is several Mbit/sec, and to cope with large amount of video connections in backbone network, required switch capacity is estimated at least at Tbit/sec. For design of switch fabric of this capacity, introduction of MIN (Multistage Interconnection Network) is indispensable.

As an appropriate type of MIN for connection oriented video traffic, Delta network[2], Benes network[3], and Clos network[4] are candidates. To maintain switch performance for large size switch, it will be desirable that switch elements have buffer within them and that arbitration of routing of timeslots are done in each switch element independently. In addition, number of stages should be small to minimize switching delay. Though Delta network has advantage among three configurations in these viewpoints, speed up of internal links is necessary to avoid internal blocking at high load rate in this network, and it will cause considerable increase of hardware. In this paper, we focus Clos network as an alternative configuration, and routing method to avoid internal blocking is proposed.

The most important feature of VTDM switch is timeslots in the same input TDM frame period are transmitted in the same output TDM frame period. Therefore, for routing method of Clos network, it will be required that all of timeslots in one input TDM frame period arrive at their destination output ports without internal blocking within corresponding output TDM frame period.

Routing methods of Clos network are classified into two groups; static routing and dynamic routing. In static routing, required characteristics for VTDM switch described above can be easily achieved if appropriate connection admission control and traffic shaping is adopted. On the other hand, in dynamic routing, there is also an existing scheme [6] to meet the demand which is designed for ATM. This method works in framed ATM switching system [7], which has similar switching principle to VTDM switch architecture, it can be simply applied to a VTDM switch.

For both of these types of routing, internal buffer is necessary in a switch element of the first stage since timeslots that are routed to the same middle stage switch should be queued to avoid blocking. Therefore total switching delay of Clos network is 1.5 times larger than that of Delta network, which is not appropriate for real-time traffic.

If routing method is devised so that only one timeslot is routed to each switch of middle stage at the same time, no buffer will be needed in switches of the first stage, and then switching delay of Clos network will be the same as Delta network.

In this paper, a new routing method in Clos network of VTDM switch to satisfy this condition is proposed. Since the proposed method is devised so that only one timeslots is routed to each switch of middle stage at the same time, and internal buffer in switches of the first stage is not required.

In the rest of this paper, operation of the proposed method is described and feasibility of the method is discussed by evaluation of processing time.

2. VARIABLE BIT RATE TDM SWITCH ARCHITECTURE

In this section, features of VTDM switch architecture are described as background of the research.

2.1 Overview of video-specific network

Figure 1 shows overview of video-specific network located in multimedia backbone network where VTDM switch architecture is applied. In this configuration, backbone network consists of traffic-specific networks for voice, video, and data communication and access network accommodates various types of traffic by integrated TDM or WDM system and provides a unified network interface to users.

2.2 Frame configuration

VTDM frame format is designed as shown in Figure 2, assuming 622 Mbit/sec transmission. To enable flexible time slot assignment for variable bit rate video traffic, header information is added at the top of the frame. Frame header has information for each time slot about connection number, priority level, and tolerable delay etc.

To perform efficient priority control for video traffic, length of VTDM frame is 1/300 (sec), which is a common divisor of frame time interval of NTSC video signal 1/30 (sec) and PAL video signal 1/25(sec).

By one timeslot is assigned for one connection for every TDM frame, the connection gets bandwidth of $(16 \text{ (kbits)} * 300 \text{ (frame/sec)}) = 4.8\text{Mbps}$

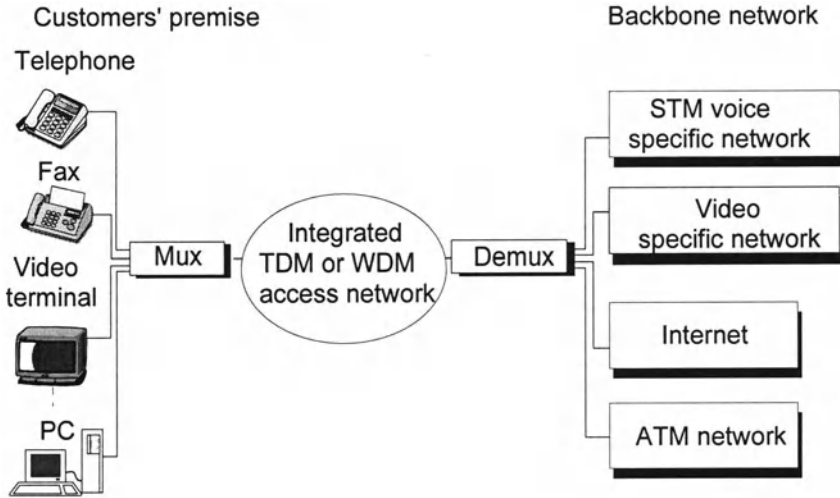


Figure 1. Overview of multimedia network system consisting of traffic-specific networks

2.3 Switch architecture and operation

There are various types in VTDM switch configuration. Figure 3 shows crosspoint buffer type configuration. At each crosspoint, a buffer and an address filter (AF) are provided. An address filter reads frame header of input TDM frame and timeslots to be routed in corresponding output port is stored in the crosspoint buffer. At each output port, timeslot scheduler and content analyzer are provided and these perform priority control according to video data contents of timeslots. When all timeslots are stored in crosspoint buffers, output TDM frame is generated by frame generator, and timeslots are assigned in the frame and sent to the next switch stage.

To guarantee timeslot switching delay, each crosspoint buffer has double buffer structure shown in Figure 4, where the length of each buffer is the same as TDM frame length. all timeslots of input TDM frame are stored in buffer (a) or buffer (b), and they are transmitted in the following TDM frame. Timeslots that cannot be transmitted at that time are discarded inevitably.

In this operation, once several timeslots are assigned in the same TDM frame at the access network of source terminal, these always arrive at a receive terminal in the same TDM frame as shown in Figure 5. That is, end-to-end delay jitter is bounded within TDM frame.

Denoting TDM frame length by T , and delay jitter by D , the following condition is satisfied.

$-T < D < T$

This delay jitter boundary is constant regardless of offered load to a switch, and it is effective for switching real time traffic including video traffic.

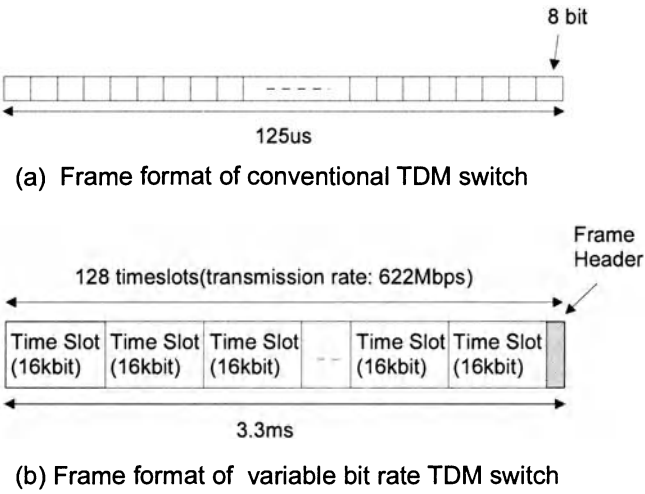


Figure 2. VTDM frame format

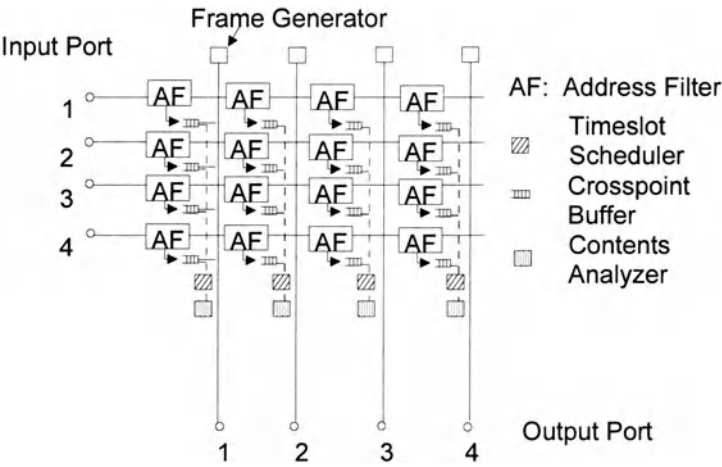


Figure 3. Crosspoint buffer type switch

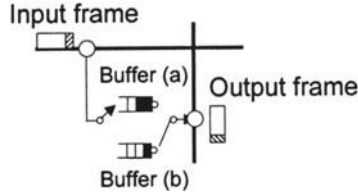


Figure 4. Double buffer structure

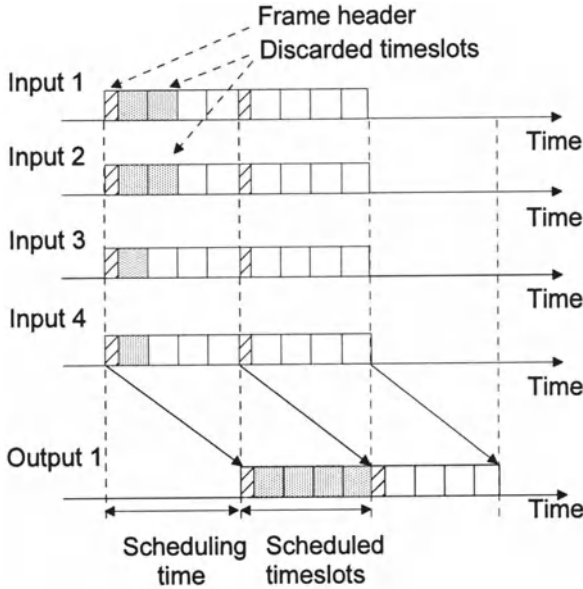


Figure 5. Timechart of switch operation

3. PROPOSAL OF A NEW ROUTING METHOD OF CLOS NETWORK

According to reasons described in Section 1, we focus Clos type network as a target to be examined, and design of large scale switch of VTDM switch using multistage interconnection network is examined.

3.1 Target configuration

In this paper, a configuration in Figure 6 is examined, where number of row and size of switch element is the same. To realize switch fabric whose capacity is Tbit/sec by VTDM switch where port speed is 622 Mbps, size of

switch must be at least 4096 x4096. Corresponding with this size, size of switch element must be at least 64 x 64.

3.2 A proposed routing method

In Clos network of VTDM switch shown in Figure 6, there is possibility that some timeslots are applied to one switch of the first stage and they are destined to one switch of the third stage. An aim of the proposed routing algorithm is that internal paths for these timeslots are distributed among all switches of the second stage to avoid internal blocking. To realize this characteristics, each switch of the first stage has a middle stage switch assignment matrix, and records results of middle stage switch assignment for timeslots in the past time. For example, usage of a matrix of the switch 11 is explained in Figure 7. Each column of this matrix corresponds with a switch of the third stage to which an input timeslot goes. On the other hand, each row corresponds with middle stage switch to which the timeslot is routed. The way to give numbers to switch elements is as shown in Figure 6. For example, if a timeslot is going to the third stage 31, and this timeslot is routed to middle stage switch 21, then the switch 11 increases a value of an element of (1,1) of the matrix by one.

When a new timeslot is applied to this switch, the switch refers all elements of a column that correspond with a switch of the third stage to which a timeslot goes. Then a middle stage switch to be routed . i.e. an output port from the first stage switch is assigned so that all values of elements of this column are statistically equal to achieve paths distribution.

Here an operation of the method is described for an example of timeslot pattern to the switch 11 shown in Figure 8. It is assumed that middle stage switch assignment for timeslots in the first timeslot position number is already finished and we focus on processes for timeslots in the second timeslot position number. At this time, initial status of the matrix before assignment is done is as shown in Figure 9 (a).

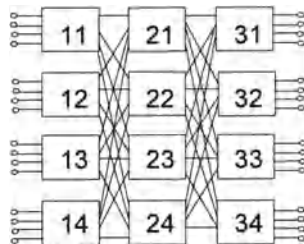


Figure 6. A configuration of Clos network examined in this paper

		1	2	3	4	Column number
		31	32	33	34	Corresponding number of switch of the third stage
1	21	0	0	0	0	
2	22	0	0	0	0	
3	23	0	0	0	0	
4	24	0	0	0	0	

Row number	Corresponding number of routed middle stage switch

Figure 7. A middle stage switch assignment matrix of the switch 11

- Phase 1 Selection of candidates of middle stage switch to be assigned

When a new timeslot is applied in a switch element of the first stage, the switch element refers elements of one column of a middle stage switch assignment matrix. This column corresponds with a switch of the third stage to which a timeslot goes. By comparing values of the elements of the column, rows in which minimum value are written are searched. Then middle stage switches that correspond with these rows are selected as candidates to be assigned to the timeslot.

For example, in case that candidates are selected for a timeslot whose ID number is (1,2), the second column of the matrix of Figure 9(a) is referred since the timeslot goes to switch 32.

- Phase 2 Assignment of middle stage switch

a) Step 1

Among all of applied timeslots to the switch, numbers of candidates of middle stage switch are compared with each other. Then a timeslot that has the least number of candidates is selected. In this case, two timeslots whose ID numbers are (1,2) and (2,2) have the least number of candidates. An order of assignment is determined as shown in the fourth column of Figure 9(b).

b) Step 2

According to the order determined in Step 1, a middle stage switch is assigned for timeslots.

c) Step 3

Going back to Step 1.

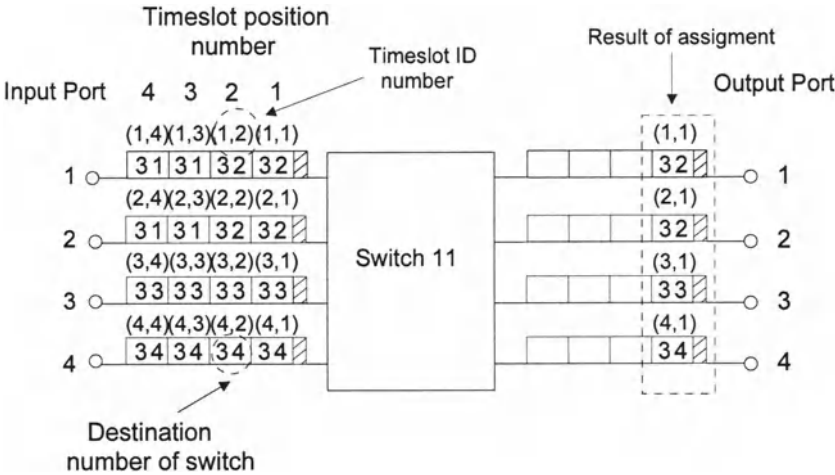


Figure 8. Timeslot input pattern to the switch 11

In this method, a timeslot that has fewer candidates of middle stage switches to be routed has priority about assignment of middle stage switch. In this way, probability that middle stage switch is assigned among candidates is improved, and efficiency of paths distribution is enhanced.

4. PERFORMANCE EVALUATION

In Clos network of VTDM switch, proposed middle stage switch assignment algorithm requires substantial amount of processing time, and a sequence of these processes must be finished within a VTDM frame interval. To demonstrate feasibility, estimation and quantitative evaluation of processing should be performed. To prepare for these evaluations, following notations are defined on Clos network of VTDM switch.

- N: number of port of Clos network
- n: number of port of a switch element of Clos network
- h: number of maximum timeslots in a TDM frame

Between N and n, we assume relationship of $N = n^2$.

	31	32	33	34
21	0	1	0	0
22	0	1	0	0
23	0	0	1	0
24	0	0	0	1

(a) State of the matrix of the switch 11

Timeslot ID number	Destination switch	Selected Candidates			Order of assignment	Assigned number of middle stage switch
(1,2)	32	23	24		1	23
(2,2)	32	23	24		2	24
(3,2)	33	21	22	24	3	21
(4,2)	34	21	22	23	4	22

(b) Assignment of middle stage switch

Figure 9. Middle stage switch assignment for timeslots in the second timeslot position

4.1 Estimation of order of amount of process

We define $C(nh)$ as amount of process of middle stage switch assignment for all timeslots applied in the switch in one TDM frame period. Moreover, we define $C(n)$ as amount of process for middle stage switch assignment for timeslots that belong to the particular timeslot position number. There are relationship of $C(nh) = hC(n)$. In this section, estimation is done for $C(n)$ in phase 1 and phase 2 respectively.

At first, In phase 1, following two processes are required to select candidates.

1. A process to select minimum value among elements of one row of the matrix.
2. A process to compare minimum value with other elements, and to select candidates for middle stage switches to be assigned.

For one of applied timeslots, both of two processes require processing time that is proportional to size of switch element. Therefore, when n timeslots are applied to a switch element, order of amount of process is calculated as $O(n^2)$.

In phase 2, sort process is also required to compare numbers of middle stage switch candidates of timeslots that are selected in phase 1. Whenever

middle stage switch is assigned to one timeslot, the sort process requires amount of process which has order of $O(n)$. Therefore, sum of amount of process of this sort for all timeslots in one timeslot position number is $O(n^2)$.

From these analysis, order of $C(n)$ is $O(n^2) + O(n^2) = O(n^2)$.

4.2 Quantitative evaluation

The amount of process of the proposed routing algorithm is also evaluated quantitatively by simulation. A routing algorithm is described by C language. A hardware platform to run simulation is Sun enterprise server (Processor: UltraSparcII processor, 300Mhz). Processing time of the algorithm is measured by internal function of UNIX named `gettimeofday()`.

The result is shown in Figure 10. Processing time is measured under condition that offered load is 100%. In the figure, both of measured values and approximation line are shown. Though this approximation line is polynomial function, it is expected to be approximated by linear function where number of ports of switch element is larger than 100, and this result shows that estimation in section 4.1 is adequate. To use this algorithm in an actual system, processing time must be less than TDM frame time of 3.3ms. In this condition of simulation, processing time is still over 3.3ms at any size of switch element. However, reduction of amount of process of the proposed method will be expected by introduction of parallel processing. For example, it is known that amount of sort process of n elements is reduced from $O(n)$ to $O(\log n)$ by introduction of parallel processing. Effects of reduction of phase 1 and phase 2 by introduction of parallel processing is calculated as follows.

– Phase 1

Sort process for each timeslot can be processed in parallel, and sort process itself can be also processed in parallel. Therefore, order of amount of process is reduced from $O(n^2)$ to $O(\log n)$.

– Phase 2

Sort process to decide to which timeslot middle stage switch is assigned can be processed in parallel. Therefore, order of amount of process is reduced from $O(n^2)$ to $O(n \log n)$.

Total amount of process will be dependent on ratio of amount of process between phase 1 and phase 2, and here it is estimated as $O(n \log n)$.

When N is 16 and n is 4, amount of sort process will be reduced by 1/2 of the original, and absolute processing time will be within 3.3ms. Switch element of 4x4 is about one twentieth of a target size of switch element of 64x64 described in Section 3. To reduce processing time farther, both of introduction of parallel processing in algorithm and devices in hardware

logic to realize this algorithm will be indispensable. The latter subject is a future work.

5. CONCLUSION

In this paper, a new dynamic timeslot routing algorithm in Clos network is proposed as an appropriate method for variable bit rate TDM switch. The most important feature of the method is that timeslot routing in switches of the first stage can be controlled without internal blocking, and operation to realize this characteristics is explained in detail. Processing time of the proposed method is evaluated by software simulation, and it is indicated that it can be controlled in real-time when switch element size is 4×4 . As a future work, design of specific hardware to reduce amount of process is planned.

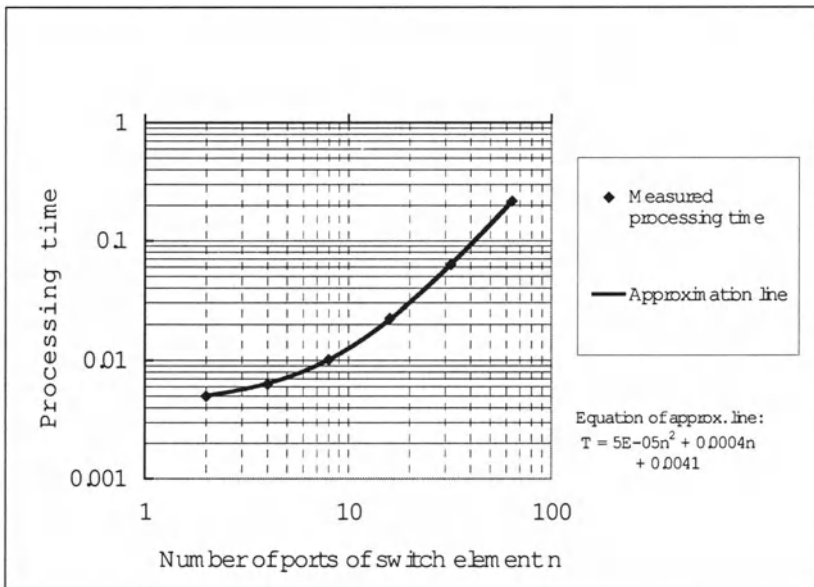


Figure 10. Evaluation of processing time by software simulation

REFERENCES

- [1] Tadao Saito, Hitoshi Aida, Udomkiat Bunworasate, Takayuki Muranaka, Terumasa Aoki. (1999). VTDM: A variable bit rate TDM switch architecture for video stream. *IEEE GLOBECOM '99*.

- [2] Patel J. (1981) Performance of processor –memory interconnections for multiprocessors. *IEEE Trans. Comput*, 70, (10), pp.771-780.
- [3] V. E. Benes. (1964). Optimal rearrangeable multistage connecting networks. *The Bell System Technical Journal*, Vol.43, No.4 pp.1641-1656.
- [4] C. Clos. (1953). A study of non-cloking switching networks. *The Bell System Technical Journal*, Vol.32, pp. 406-424.
- [5] Satoru Ohta, and Haruo Yamaguchi. (1987). A Non-Congestion Self-Routing Control Algorithm of Packet Switching Networks. *Transactions of IEICE J70-A*, 2, pp.312-319. (In Japanese).
- [6] Hideki Satoh, Yoshihiro Ohba. (1992). Switching Algorithm and Network Control in Framed ATM Networks. *Workshop of IEICE SSE92-22*. (In Japanese).
- [7] S.J.Golestani. (1991). Duration-Limited Statistical Multiplexing of Delay-Sensitive Traffic n Packet Networks. *IEEE INFOCOM '91*, pp.4A.4.1-4.A.4.10.

ATM NETWORK SECURITY

Requirements, Approaches, Standards, and the SCAN Solution

Herbert Leitold, Reinhard Posch

Institute for Applied Information Processing and Communications (IAIK)

Graz University of Technology

Inffeldgasse 16a, A-8010 Graz, Austria

Herbert.Leitold@iaik.at

Reinhard.Posch@iaik.at

Key words: ATM security, ATM confidentiality, Key agile encryption

Abstract: Broadband networks based on the asynchronous transfer mode (ATM) are emerging rapidly. Both the technological component in terms of ATM infrastructure, as well as the area of applications requiring Quality of Services (QoS) by the means of bandwidth or delay constraints are covered by a variety of projects and products. However, given the increasing interest in applications such as governmental communication, transmission of medical information, or commercial applications, the necessity of providing secure means of delivering sensitive contents is apparent.

In this paper, we focus on security services in ATM networks. The variety of different approaches and solutions are categorised by the means of its common and distinct functionality, as well as certain advantages and disadvantages are discussed. In addition, the standardisation efforts by the leading group in that area—the ATM Forum—are outlined. Finally, the essentials of the project SCAN³ are given, resulting in a comprehensive solution to security services in ATM networks.

³ The project Secure Communication in ATM Networks (SCAN) is funded by the European Commission, Directorate General XIII under the Advanced Communications Technologies and Services (ACTS) Programme, project number AC330.

1. INTRODUCTION

Broadband networking has evolved dramatically in the past few years. Especially the asynchronous transfer mode (ATM) has attracted attention in the area of high-speed communication, promising the integration of almost all communication profiles such as voice, video and computer-based multimedia communication. Having had its first broad deployment in the high-speed local area network (LAN) arena [1], the emerging deployment of ATM in the public wide area network (WAN) sector is going on, such as the Trans-European Network 34 Mbps (TEN-34) [2] and its 155 Mbps successor TEN-155. ATM builds the basis of broadband integrated services digital network (B-ISDN) and is to be considered as one of the key technologies of the data-highway initiatives aimed by the majority of industrial countries, as described by the User Requirements and Strategies for Application (URSA) report [3]. Beside the capability of carrying the variety of communication profiles, the importance of ATM in the broadband networking arena is emphasised by the fact that it is defined basically independent from the physical media. Consequently, as surveyed in [4] and [5], promising residential access methods, such as asymmetric digital subscriber line (ADSL), or fibre-coax hybrid cable television (CATV) networks are focused towards ATM, as expressed by Maxwell for ADSL [6], respectively by Hernandez-Valencia for cable-networks [7].

However, as soon as publicly accessible infrastructure is involved, security is a major concern. ATM distinguishes from other networking technologies by the means, that the need of investigating security services was recognised early, in fact prior to a broad deployment of ATM networks. Both the research community, and standardisation bodies have been the driving factors: On the one hand, a number of approaches to security in ATM networks have been published by scientists. On the other hand, the leading groups working on ATM standardisation—the ATM Forum—established a security working group in 1996 [8], resulting in the ATM Forum Security Framework [9], as well as the ATM Forum Security Specification 1.0 [10] which has been approved recently in February 1999.

In this paper, we discuss these different approaches. Therefore, the remainder of this paper is structured, as follows: Section 2 describes the basic security requirements given in ATM networks. This is followed by the discussion of different approaches to ATM security in section 3. The ATM Forum Security Specification 1.0 is described in section 4. Section 5 outlines the project Secure Communication in ATM Networks (SCAN) that implements above stated specification by the means of a secure ATM network interface card (ATM-NIC) and, finally, conclusions are drawn.

2. ATM SECURITY REQUIREMENTS

ATM is a cell relay technique operating on protocol data units (PDUs) of a fixed size, called ATM cells. Similar to other packet oriented networks, such as X.25 or Frame Relay, ATM is a connection oriented technology by the means of virtual channels (VCs). Using signalling procedures VCs are established on demand as switched virtual connections (SVCs), or as permanent virtual connections (PVCs).

Closely related to the integrated services digital network (ISDN) protocol reference model, the ATM respectively the B-ISDN protocol architecture has been standardised by the International Telecommunication Union, Telecommunication Standardisation Sector (ITU-T) in recommendation I.321 [11]. It consists of several planes: Firstly, the management plane to maintain the network operational. Secondly, the control plane defines the signalling functions needed to establish and control the communication channels (VCs). Finally, the user plane provides transfer of user data across ATM virtual channel connections (VCCs) or virtual path connections (VPCs). The ATM protocol reference model is illustrated in figure 1.

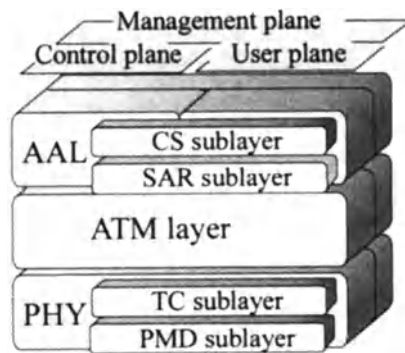


Figure 1. ATM (B-ISDN) protocol reference model

In addition to the planes, the ATM protocol reference model consists of several layers: At the top of the hierarchy the ATM adaptation layer (AAL) converts native ATM data or non-ATM information to the ATM cell format. Therefore, convergence sublayer (CS) and segmentation and reassembly (SAR) functions are employed. The ATM layer operates on ATM cells which consist of a five octet header used for guiding the information through the ATM network and a 48 octet payload that carries the user information. Finally, the physical layer embeds the ATM cell to the transmission frame, such as a synchronous digital hierarchy (SDH) frame. Therefore,

transmission convergence (TC) and physical media dependant (PMD) functions are employed.

An ATM network consists of a set of ATM switches interconnected by point-to-point physical links. Two different types of interfaces are provided: Firstly, the user network interface (UNI) [12] between an user and an ATM switch, respectively between two ATM switches in some cases. Secondly, the network-network interface (NNI) [13] is defining the routing and signalling mechanisms needed within larger clouds of ATM switches. In addition, the UNI is subdivided into public-UNI and private-UNI, depending on whether the ATM link is connected to a public or private ATM switch.

From the security point of view an ATM network can be looked upon as a set of endpoints (users) interconnected by a set of ATM switches. Thus, the security interactions can be characterised as an user-to-user, an user-to-network, or a network-to-network interaction scenario. This is schematically depicted in figure 2. Two VCs are shown, VC_1 between end system A and end system C, and VC_2 between end system B and end system C.

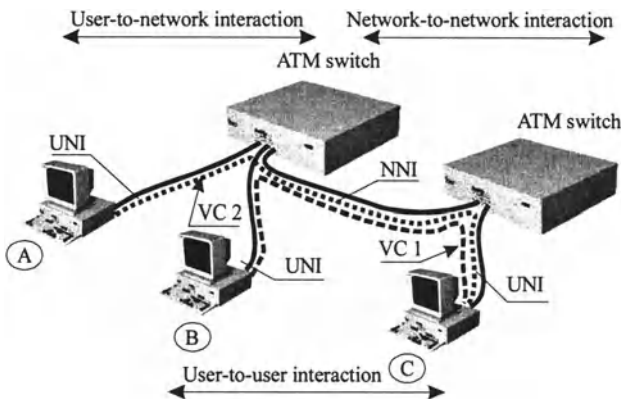


Figure 2. ATM security interaction model

With reference to the protocol model shown in figure 1, in particular the tasks assigned to the three planes, the security requirements are to be identified as follows:

- **User plane:** In the user plane providing transfer of user data across ATM VCCs or VPCs, four security requirements can be identified: Authentication, data confidentiality, data integrity, and access control. With reference to figure 2, these security requirements refer to the user-to-user interaction model.
- **Control plane:** The control plane performs connection establishment and connection maintenance functions. Both the user-to-network interaction model and the network-to-network scenario—as shown in figure 2—are

of importance. Since signalling message fraud affects the state and the availability of the network, the protection of control plane messages is of vital importance. Consequently, authentication and integrity are the basic security requirements.

- **Management plane:** The management plane performs management and coordination functions related to both the user, and the control plane. So far, ATM management plane security has not been that investigated in the literature. However, at least authentication and access control can be identified as security requirements.

It is worth mentioning that all approaches appearing in the literature have above listed requirements in common (see also [9]). Apart from the security features related to the different planes, a common functionality are the so-called support services, assisting the security services by providing key exchange, key update mechanisms, public key infrastructures, and further security parameter negotiation.

Having identified the basic security requirements, the following section 3 surveys the different approaches to security in ATM networks. Guided by a qualitative discussion of the advantages and the disadvantages of a certain class of approaches, the common features are discussed.

3. ATM SECURITY APPROACHES

In order to categorise the multitude of proposed and established approaches to ATM security, this section assigns the approaches to the protocol layer the cryptographic mechanisms are applied to: Subsection 3.1 discusses application layer security services, subsection 3.2 addresses AAL based approaches, subsection 3.3 investigates embedding of security services into the ATM layer, and PHY layer based approaches are discussed in subsection 3.4. Finally, support services are addressed in a separate subsection 3.5, as these services are rather related to the planes of the ATM protocol reference model, than to a certain layer.

3.1 Application layer security services

The first approach appearing in the literature is applying security services to the protocol layers located above the ATM protocol model. Such an approach, for instance integrating the security services into the network layer of the open system interconnect (OSI) model (e.g. Internet protocol IP), the OSI transport layer (e.g. Berkeley Software Distribution BSD sockets), or even the application itself (e.g., a secure hypertext transport protocol HTTPS

server), makes the approach independent from ATM and, apart from the high transmissions rates in the multi-Mbps range offered by ATM, gives no substantial difference to well understood methods, such as IPSEC or secure socket layer (SSL).

However, this class of approaches is addressed in this paper to identify the major drawback: Due to the high bandwidth required, integrating cryptographic algorithms into applications fails for performance reasons. Hardware support is needed, hard to achieve at the protocol layers involved in a platform and service independent manner, as these protocols are usually implemented in software.

As one representative of this class of approaches, [14] is referenced: The work described by Ellermann and Benecke addresses the performance when introducing conventional firewalls into high-speed ATM networks.

3.2 AAL security services

The first class of approaches integrating security services into the ATM protocol model is applying them to the AAL. Especially for computer based traffic, this approach is alleged advantageous, as one certain AAL (AAL-5) is mainly used. The following figure 3 illustrates the two interfaces where security services can be integrated into the AAL. These are the interface between the two AAL sublayers CS and SAR which is shown as (a) in figure 3, respectively the interface between the AAL and the host computer's protocol stack which is shown as (b) in figure 3.

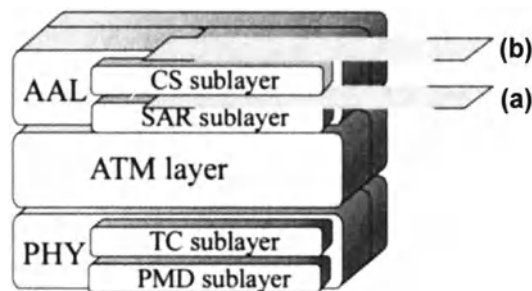


Figure 3. AAL security services embedding

AAL embedding of security services has been proposed several times. The main representatives of this approach are, as follows:

- General AAL approaches: For user plane confidentiality, Cruickshank, et. al proposed to encrypt the AAL-SDU payload [15]. A similar attempt to integrate security services into the AAL is described by Forné and Melús [16].

- CS-SAR interception: In the approach discussed by Geng, et. al. [17], the security services intercept the interface between the CS sublayer and the SAR sublayer of the AAL by introducing a data protection layer (DPL). In this approach, user plane confidentiality and user plane integrity are provided.
- Host-AAL interception: An alternative has been proposed by Chuang in [18], where a CryptoNode intercepts AAL frames at the direct memory access (DMA) interface between the host computer and the ATM-NIC. Targeting encryption of the user data, user plane confidentiality is provided in that approach. Comparable to this is a Solution for Frequent Communication (SFFC) given by Laurent, et. al. in [19], where the user plane confidentiality service is also layered above the AAL.

Above stated approaches have in common that the aim is to secure computer based communication in ATM network infrastructures. However, considering its implementation, two major drawbacks can be identified: On the one hand, the service independence ATM offers is counteracted, as the approaches are limited to that certain service class (AAL). On the other hand, at the high data rates to be supported the cryptographic algorithms are not feasible in a software implementation and state-of-the-art components do not offer the required hardware interfaces, except the DMA interface in the host-AAL interception approach described. In particular, user plane confidentiality using strong encryption algorithms requires hardware implementation of the cryptographic units.

Thus, it is to be concluded that, for user plane confidentiality, placing the security services in the AAL limits the flexibility in terms of service independence and faces implementation problems. However, for providing data integrity, integration into the AAL is advantageous, as the performance required for the message authentication codes (MAC) is feasible in software. In addition, the MAC can be transparently added to the AAL-SDU, whereas this is problematic in the ATM layer due to the fixed size of ATM cells.

To cover that conflicting advantages and drawbacks, Cherukuri, Peyravian, and Wu conclude to integrate data integrity into the AAL, whereas data confidentiality is best located in the ATM layer [20]. This ATM layer approach is described in the following subsection 3.3.

3.3 ATM layer security services

Given the drawbacks regarding implementation concerns as described in the previous subsection, embedding data confidentiality into the ATM layer is advantageous. This makes the solution independent from the AAL and, thus, independent from the service class. In this approach, the 48 octet ATM

cell payload is encrypted, by the way neatly fitting the block size of many symmetric block ciphers, such as the eight octet block size of the data encryption standard (DES) [21]. The five octet ATM cell header carrying the information to route ATM cells through the network is kept unchanged. Regarding embedding the cryptographic functions into the ATM protocol model, two opportunities exist:

- AAL-ATM interception: In [20], intercepting ATM cells at the interface between the AAL and the ATM layer is proposed. This is sketched as (a) in figure 4.
- ATM-PHY interception: As state-of-the-art very large scaled integrated circuits (VLSI) used for ATM end devices integrate both the ATM layer, and the AAL, access to the AAL-ATM interface is not given in many cases. An alternative is proposed in [22], where the interface between the ATM layer and the PHY layer is intercepted, sketched as (b) in figure 4. This interface is standardised as the universal test and operations physical interface for ATM (UTOPIA). UTOPIA is supported by almost any ATM component manufacturer. The approach is referred to as UTOPIA-intercepting encryption.

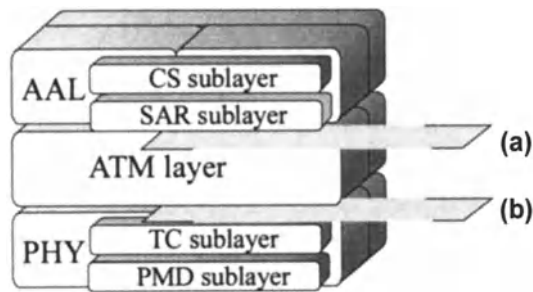


Figure 4. ATM cell level security services

Compared to AAL embedding, ATM layer embedding of security services has its main advantages for concerns of service independence when applying encryption, as stated above. However, beside the high data rates to be dealt with, in fact the same as in the AAL approach, an additional challenge is introduced: Due to the small cell payload, together with the fact that ATM cells assigned to different VCs arrive statistically multiplexed, the session keys used for encryption are to be changed rapidly, if unique keys are used for each VC. This has been termed key agile encryption by Stevenson, et. al. [23]. A further discussion of the key agility problem employing content addressable memory (CAM) techniques is given in [22].

3.4 PHY layer security services

Security services can be embedded into the PHY layer. This has been proposed by Rieke in [24]. In that approach providing data confidentiality, the whole ATM cell including the header information is encrypted. However, as the header information is to be interpreted by the intermediate ATM switches, as well as the virtual path identifier (VPI) virtual channel identifier (VCI) information is translated whilst passing an ATM switch, this approach requires decryption and re-encryption at each intermediate node. The user perceptive of endpoint-to-endpoint confidentiality in the user-to-user interaction scenario (see figure 2) inhibits this approach.

3.5 Support services

We complete this section by discussing how support services to perform key exchange or authentication protocols can be embedded into the ATM protocol model.

To perform authentication or to negotiate session keys, a transparent information channel is required. Three approaches exist, one related to the control plane, one to the user plane, and one employing management cells:

- Control plane information elements: Upon establishment of a VC, information elements (IE) carry the signalling data. The first approach to provide a support services communication channel is to enhance the signalling IEs by specific security IEs. This has the advantage of embedding the support services into the call setup phase and, thus, keys can be negotiated prior to the completion of VC establishment.
- User plane blocking: The disadvantage of additional signalling IEs is, that the ATM switches have to accept and transparently forward this additional information. An upgrade of the ATM network is required. In addition, the UNI version 3.1 does not provide sufficient information flows to perform three-way session key negotiation. A workaround is, to exclusively assign the VC to the support services for the period required for session key negotiation immediately after a VC has been established. Thus, the user plane is blocked for the application in that period.
- OAM flows: To re-negotiate session keys during a call, in-band operation and management (OAM) cells can carry the data. This is done using F4 flows on the VPC level, the VCC level uses F5 flows.

This completes the discussion of approaches to ATM network security. The approaches mainly address user plane confidentiality, data integrity, and authentication. In the following section 4 we continue in describing the ATM Forum Security Specification as a standards framework.

4. ATM FORUM SECURITY SPECIFICATION

The ATM Forum Security Specification 1.0 [10] has been approved in February 1999. With reference to the basic security requirements identified in section 2, the essentials of the ATM Forum approach are to be summarised, as follows:

- Entity authentication: Entity authentication is defined for the user plane. Either symmetric algorithms or asymmetric algorithms can be used.
- Data origin authentication and integrity: This is defined for the user plane and the control plane: In the user plane, MACs such as DES cipher block chaining (DES/CBC-MAC) are appended to the AAL-3/4 or the AAL-5 SDU. In the control plane this is accomplished between adjacent signalling elements by applying the user plane approach to the signalling channel.
- Data confidentiality: For confidentiality in the user plane, the ATM Forum follows the ATM layer approach, as described in section 3.3. Symmetrical encryption algorithms are defined, namely DES, TripleDES and FEAL in cipher block chaining (CBC), electronic code book (ECB), and a counter operational mode [25].
- Access control: User plane access control is defined on a per-VC basis using security labels. Access control is performed during connection establishment based on the security message exchange
- Support services: All three methods described in section 3.5 are defined, namely security IEs, user plane blocking, and using OAM flows. In addition, the security message exchange mechanisms provide for the exchange of X.509 public key certificates.

The ATM Forum Security Specification 1.0 constitutes a standards framework that enables interoperable solutions to secure communication in ATM networks. In the following section 5, we present a solution which results from a project the authors of this paper have been involved in.

5. THE SCAN PROJECT RESULT AND FINDINGS

The survey presented origins in the project SCAN, a project embedded into the European Commission Advanced Communications and Services (ACTS) Programme. The SCAN project objective is to provide confidential communication in ATM networks in an approach independent from the application, as well as the physical media ATM is applied to.

Streamlined to the ATM Forum Security Specification 1.0 [10], the project SCAN follows the UTOPIA interception approach described in

section 3.3. An VLSI DES/TripleDES encryption unit intercepts the UTOPIA interface between the TC controller and the SAR processor of an ATM NIC. Note, that the term “*SAR processor*” refers to the VLSI integration of the ATM layer and the AAL common in state-of-the-art ATM end system components, as stated in section 3.2. The described scenario is illustrated in figure 5.

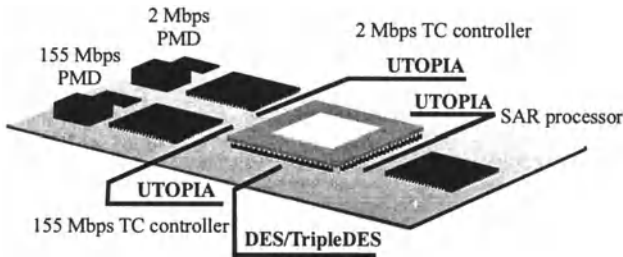


Figure 5. UTOPIA intercepting ATM cell payload encryption

As shown in figure 5, two physical interfaces are provided: On the one hand, the 155 Mbps synchronous transfer mode 1 (STM-1) constitutes the top level throughput. On the other hand, a 2 Mbps E1 interface is shown. This independence from the physical media is to be considered one of the main advantages of the project, as the variety of different media ATM is defined for is supported. Thus, the approach is also considered for application to satellite networks [26] and ADSL [27].

Regarding user plane confidentiality, the core unit is the key agile high-speed DES/TripleDES encryption unit (HADES). This VLSI complementary metal oxide semiconductor (CMOS) unit has been specifically designed for ATM user plane confidentiality requirements. It offers two UTOPIA compliant interfaces, one to the ATM layer component substituting a TC controller and one to the PHY layer component substituting a SAR processor. Consequently, the components interfacing to HADES are fooled a conventional ATM NIC architecture and, thus, HADES is transparent to both the SAR processor, and the TC controller, as illustrated in figure 5. Note, that figure 5 shows the two blocks 155 Mbps and 2 Mbps for concerns of illustrating the physical media independence, although just one of the two blocks can be the active one.

As ATM cells pass the encryption unit, HADES extracts the VPI/VCI pair, as well as the payload type identifier (PTI) from the ATM cell header. Spoken in exact ATM protocol model terminology, the model operates below the ATM layer. However, as the UTOPIA interface provides a start of cell (SOC) signal, the ATM cell header can be easily reconstructed from the

octet stream passing the eight bit UTOPIA busses. Thus, the approach is to be considered as an ATM layer embedding of the encryption process.

The VPI, VCI information identifies the VC, whereas the PTI information identifies the ATM cell type. In the per-VC encryption approach followed, which means that each user connection (each VC) is assigned a unique session key, HADES interprets the $\langle VPI, VCI, PTI \rangle$ triple as the encryption context applied to the ATM cell payload. This can be an arbitrary combination of DES, TripleDES, and plaintext communication, respectively ECB and CBC operational mode for each VC and communication direction transmit or receive. Providing arbitrary combinations of encryption algorithms (DES or TripleDES) and operational modes (ECB or CBC) is referred to as algorithm agile encryption [28], robustness agile encryption respectively [29]. HADES employs on-chip session key storage and management blocks which utilise CAM techniques to identify a certain encryption context, and random access memory (RAM) techniques to store the session keys and the initialisation vectors (IVs). The CAM techniques used allow to identify the encryption context within the period the first eight octet ATM cell block is loaded via UTOPIA. This minimises the delay introduced by the encryption process to below one ATM cell period. This is worth mentioning, as this minimal additional delay does not affect the QoS parameters of the application. Actually, the latency between ATM cell data entering HADES at the input UTOPIA interface and the corresponding encrypted data leaving HADES at the output UTOPIA interface is in the magnitude of 1 μ s when operating at the top level throughput of 155 Mbps which is significantly below an ATM cell period (an ATM cell period is approximately 2.8 μ s at 155 Mbps STM-1).

Whereas the encryption services are implemented in hardware giving the flexibility of a robustness agile ATM encryptor, a higher degree of flexibility is required for the security context negotiation process including session key exchange and authentication. Specific user requirements, as well as legal constraints may require different key exchange methods applying different cryptographic strength in terms of key length. To decouple the encryption unit from the key exchange process, the corresponding functions are implemented in software and embedded into the operating system's kernel space, included into the drivers section of the SCAN NIC. The session keys are enclosed into specific OAM F5 section flows and, thus, ATM cells passing the UTOPIA interface are used to load an encryption context to HADES. This keeps the hardware and platform independence of the encryption unit.

The overall objective of SCAN is a secure ATM NIC fitting a single-slot peripheral component interface (PCI) bus for PCs. The drivers implemented are network driver interface standard (NDIS) version 5 supporting

Windows 2000 and Windows 98. The secure ATM NIC provides user plane confidentiality by employing HADES featuring DES and TripleDES in CBC and ECB operational mode. The physical interfaces supported are 155 Mbps STM-1 and 2 Mbps E1. End system authentication and session key exchange using 1024 bit RSA, as well as the support services utilising security IEs for UNI version 4.0, user plane blocking, and OAM flows as discussed in section 3.5 are integral part of SCAN. At the time of writing this paper, the design of HADES has been completed in a 0.6 micron CMOS process. First engineering samples and, thus, prototypes of HADES integrated into the SCAN secure ATM NIC are expected for the forth quarter of 1999.

6. CONCLUSIONS

The paper has presented a comprehensive survey on the topic of security services in ATM networks. The security requirements have been identified and the variety of different approaches has been described. The advantages and disadvantages of each approach have been discussed. In particular, implementation issues have been identified, such as applicability to state-of-the-art ATM hardware. The standardisation efforts by the ATM Forum have been outlined and, finally, the project SCAN which is streamlined to this standard has been presented.

REFERENCES

- [1] I.F. Akyildiz and K.L. Bernhardt, ATM Local Area Networks, A Survey of Requirements, Architectures, and Standards, IEEE Communications Magazine v. 35, n. 7, 1997.
- [2] M.H. Behringer, The Implementation of TEN-34, Proceedings of 8th Joint European Networking Conference JENC'97, 1997.
- [3] URSA consortium, User Requirements and Strategies for Application, Final report of RACE project URSA, R2091, WP7, 1995.
- [4] B. Khasnabish, Broadband to the Home (BTTH): Architectures, Access Methods, and the Appetite for it, IEEE Network v. 11 n. 1, 1997.
- [5] L.A. Ims, D. Myhre, B.T. Olsen, Economics of Residential Broadband Access Network Technologies and Strategies, IEEE Network, v. 11, n. 1, 1997.
- [6] K. Maxwell, Asymmetric Digital Subscriber Line: Interim Technology for the Next Forty Years, IEEE Communications Magazine v. 34, n. 10, 1996.
- [7] E.J. Hernandez-Valencia, Architectures for Broadband Residential IP Services Over CATV Networks, IEEE Network v. 11, n. 1, 1997.
- [8] M. Peyravian and T. Tarman, Asynchronous Transfer Mode Security, IEEE Network, v. 11, n. 3, 1997.
- [9] ATM Forum, ATM Security Framework 1.0, The ATM Forum Technical Committee, AF-SEC-0096.000, 1998.

- [10] ATM Forum, ATM Security Specification, Version 1.0, ATM Forum Technical Committee, ATM-SEC-01.010, 1999.
- [11] ITU-T, B-ISDN Protocol Reference Model and its Application, Recommendation I.321, International Telecommunication Union, Telecommunication Standardisation Sector, 1991.
- [12] G. Dobrowsky, (Ed.), ATM User-Network Interface Version 3.1 Specification, The ATM Forum, Technical Committee, 1994.
- [13] M. Goguen, (Ed.), Private Network-Network Interface Specification Version 1.0, The ATM Forum, Technical Committee, 1996.
- [14] U. Ellermann, C. Benecke, Firewalls for ATM Networks, Proceedings of International Congress on Information Technology Security INFOSECcom, 1998.
- [15] H. Cruickshank, Z. Sun, S. Valentzas, A Proposal for Security Services in ATM Networks, Proceedings of the 4th IFIP Workshop on Performance Modelling and Evaluation of ATM Networks, 1996.
- [16] J. Forné, J.L. Melús, An integrated solution for secure communications over B-ISDN, in: Communications and Multimedia Security II, ed. P. Horster, Chapman & Hall, 1996.
- [17] R.H. Deng, L. Gong, A.A. Lazar, Securing Data Transfer in Asynchronous Transfer Mode Networks, Proceedings of Globecom'95, 1995.
- [18] S.C. Chuang, Securing ATM Networks, Proceedings of 3rd ACM Conference on Computer and Communications Security, 1996.
- [19] M. Laurent, O. Paul, P. Rolin, Securing communications over ATM Networks, in: Global IT Security, ed. L. Yngstrom, Chapman & Hall, 1997.
- [20] R.J. Cherukuri, M. Peyravian, S.F. Wu, A User Plane Security Protocol for ATM Networks, Proceedings of 5th International Conference on Telecommunication Systems, 1996.
- [21] ANSI, American National Standard for Data Encryption Algorithm (DEA), ANSI 3.92, American National Standards Institute, 1981.
- [22] H. Leitold, U. Payer, R. Posch, A Hardware Independent Encryption Model for ATM Devices, Proceedings of 14th Annual Computer Security Applications Conference ACSAC'98, 1998.
- [23] D. Stevenson, N. Hillery, and G. Byrd, Secure Communications in ATM Networks, Communications of the ACM, v. 38, n. 2, 1995.
- [24] A. Rieke, Link Encryption in ATM Systems, in: Communications and Multimedia Security III, ed. S. Katsikas, Chapman & Hall, 1997.
- [25] ANSI, American National Standard for Information Systems-Data Encryption Algorithm-Modes of Operation, ANSI 3.106, American National Standards Inst., 1983.
- [26] H. Cruickshank, B.G. Evans, I. Mertzanis, H. Leitold, R. Posch, Securing Multimedia Services Over Satellite ATM Networks, International Journal of Satellite Communications, v. 16, n. 4, 1998.
- [27] E. Areizaga, P. Ibañez, H. Leitold, R. Posch, M. Laurent, J.M. Mateos, J.L.M. González, Secure Communications in ATM over Access Networks, to appear in proceedings of Broadband Access Conference BAC'99, 1999.
- [28] T.D. Tarman, R. L. Hutchinson, L. G. Pierson, P. E. Sholander, E. L. Witzke Algorithm-Agile Encryption in ATM Networks, IEEE Computer, v. 31, n. 9, 1998.
- [29] L.G. Pierson, E. L. Witzke, M. O. Bean, G. J. Trombley, Context Agile Encryption for High-Speed Communication Networks, ACM SIGCOMM, Computer Communications Review, v. 29, n. 1, 1999.

UNIFIED MESSAGING

A Framework Approach for building messaging services

Timo HäMäläinen and Esa Saukkonen

Centia Ltd., sponsored by Hewlett-Packard

Key words: Unified messaging, Value-added services, Frameworks

Abstract: This paper presents an approach for the development of a unified messaging (UM) solution, which enables users to combine their messages into a single virtual mailbox, enabling message access, response and administration from anywhere, at anytime.

1. INTRODUCTION

Competition in the mobile market has now reached the stage where falling margins from traditional basic services are impacting upon the revenues of most European operators. As price differentials between operators in national mobile markets are reduced, the introduction of carefully priced, sophisticated value added services is seen as a prime alternative revenue source and a key factor in winning and retaining customer loyalty. One service considered by most operators to be of critical importance for customers is that of message management.

Increasing competitive pressures and rising costs are forcing enterprises worldwide to improve the speed and efficiency with which they do business with their customers. Often this means equipping a mobilised work force to enable rapid communication with customers and clients from anywhere at any time but, for many organisations, this task is complicated by the variety of message services employed. The mixture of fax, voicemail and email services requires that multiple devices are required to send, respond to and administer these messages, resulting in increased maintenance costs and, frequently, in lost revenues due to late response times.

While the scope for outsourcing among Small to Medium sized Enterprises (SME's) is not so broad as for large corporations, re-organisation of the office administration and communication functions provides equally significant opportunities to enhance profitability through improved efficiency. Most SME's, however, do not have the capability to have their own messaging services because of the relatively high cost of these solutions. Because of this reason, service provider offered solution best fits for the SME sector.

This paper presents an approach for the development of a unified messaging (UM) solution which enables users to combine their messages into a single virtual mailbox, where they can be accessed and processed anywhere, at anytime.

2. REQUIREMENTS FOR UM SOLUTION

As mentioned earlier, the term unified messaging describes the nature of the resulting services, where different message types are combined into a single virtual mailbox, which can be accessed with multiple types of methods. Virtual mailbox can be achieved by collecting messages from various sources into a central location. Multiple access methods can then be obtained by connecting this 'central mailbox' into different kinds of access networks like Internet or telephony network.

It is also important to note that when there is a way to access many types of messages, messaging can be made more intelligent. This means that UM solution should provide for example ways to convert messages from one type to another and ways to notify people who don't have temporarily email client available.

2.1 A closer look at interfacing need

UM solution must interface with the various PSTN, PLMN and Internet components, which are needed to receive and deliver messages from and to other messaging systems. Some of these interfaces conform to international standards but others are proprietary systems. The following diagram details some of the more important components with which the UM solution must interface.

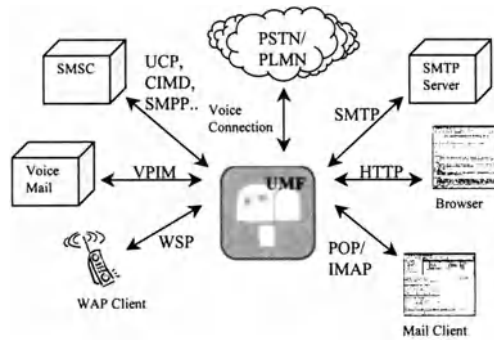


Figure 1. Interfacing with other systems

The most common messaging systems today are email and voicemail. In order to receive email messages the UM solution must support SMTP protocol, which is used between Internet mail servers to deliver mail messages, and therefore act like a normal SMTP server. The UM system must also interface with other email servers which employ POP or IMAP protocol – and thus behave like a normal email client, enabling message retrieval from other possible mailboxes of a user to a central place.

Retrieval of voicemail is much more complex issue because voicemail systems do not normally provide external interfaces. This issue will be hopefully resolved with the introduction of VPIM (Voice Profile over Internet Mail) standards, which define how to transport voice and fax messages over SMTP protocol. An alternative solution is to build voice mail capabilities into the UM system, which means that UM system must interface with telephony network.

Message notification is a critical feature for any messaging system. In the GSM environment the short messaging feature provides a convenient method to deliver notification and it can also be used to send the actual messages and to carry mailbox management information. Using of short messages can be motivated with fact that they have been used to implement many kinds of information retrieval and other value-added services in Scandinavia. Short message systems interfaces are often proprietary, the most common of which are UCP, CIMD, GPI and SMPP.

The most common way of accessing messages is an email client. Web browsers can also be used to access messages and, in the near future, access methods such as Wireless Application Protocol (WAP) can be used to run a micro web browser on mobile phones providing the necessary mailbox access. Besides of accessing messages, web browsers can be used to manage UM system configuration. Configuration management with web browser is useful especially for users to modify their private configuration information,

like passwords since browser is commonly accepted and commonly available information access tool.

3. FRAMEWORK APPROACH AS A UM SOLUTION

Today most of the value added messaging services are being built one by one resulting fast time to market for the first few services. Here, every messaging service setup is custom project, thus resulting to unacceptable service creation speed in longer term. Also, individual services require vast management resources. The framework approach offers fast time to market for new messaging services through utilisation of existing resources.

The UM solution described in this paper comprises UM services and UM framework (UMF). UM framework is a collection of components that simplify application development and run-time organisation. UM service applications together with UM framework provide the functionality which is visible to the users of UM services.

4. UM FRAMEWORK LOGICAL ARCHITECTURE

This section describes the logical architecture of the UM framework (UMF) which not only presents a unified method of accessing different kinds of messages but also offers a sophisticated platform for service providers to develop a wide range of UM services.

The following schematic outlines the architecture and the environment of the UM framework. This logical architecture is not, however, limited to unified messaging and can be applied to many kinds of network service platforms since it is not bound to specific networks or services.

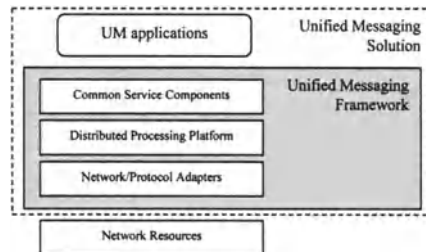


Figure 2. Architecture of the UM framework

The UMF environment comprises UM applications and network resources. UM applications access and manipulate different kinds of

messages and they can be built by harnessing the components provided by the framework. It is the UM applications which implement the unified messaging functionality that can be seen by the users of the UM services.

A typical UM application would be a 'smart mailbox with SMS notification' which is simply a mailbox, which sends a notification about an incoming email to the receiver by using a GSM short message. This application could also be employed to automatically retrieve emails from the user's other mailboxes to a central location because people usually have multiple mailboxes. When a user receives SMS notification, one can retrieve the mail from POP email server application by using POP compatible email client or use text-to-speech email application to listen the mail.

Network resources represent different kinds of networks, which can be used to access the UM applications. These resources can also be employed by the applications since their functionality can contain network communication like in the case of SMS notification described earlier. Because the unified messaging in general gives a unified view into a message box containing different types of messages from various kinds of clients, different kinds of networks and protocols must be adapted to the framework.

4.1 UM framework layers

This section describes various layers of the UM framework. The components of UMF have been categorised into layers according to their functionality:

- Distributed processing platform enables communication between the components of UMF.
- Network/Protocol adapters connect the framework to different networks.
- Common service components are reusable functional blocks which are being utilised by the UM applications.

4.1.1 Distributed processing platform

The UMF and UM applications consist of components which interact with each other. The distributed processing platform (DPP) provides support for their execution and interaction in the distributed environment. Since these are the objectives of almost all the commercial middleware products, a standard Corba compliant ORB is used since Corba supports object oriented modelling and allows distribution of the platform over a heterogeneous set of computers. Corba ORB establishes a client-server relationship between components. The client can transparently invoke an operation on a server object, which can either be on the same machine or across a network. Corba

architecture uses an Interface Definition Language (IDL) to describe the interfaces and an interface description defines the services of a component.

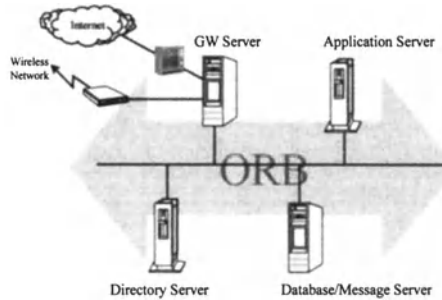


Figure 3. Distribution of UMF

The DPP also provides features, which support run time operations such as load balancing. Load balancing shares the load of the framework and the applications between several computer nodes.

By using a distributed platform such as Corba it is possible to integrate and still distribute the various components of the UMF over several network nodes. This kind of solution offers not only distribution benefits in general, but also benefits specific to UM and frameworks:

1. One of the principal drivers behind the framework concept is to provide an easy and cost effective means to increase the performance of the UM solution. Distribution of UMF and UM applications over several machines allows service providers to start with a simple configuration and a limited set of services without having to make huge investments in the UM solution.
2. The framework concept also enables customisation of a UM solution for specific environments. A small-scale framework can initially comprise only the essential functionality and a small set of services but may still be expanded to a higher performance system offering a richer set of complex services. This kind of expandability allows service providers to prepare for future messaging access methods such as WAP and UMTS.

4.1.2 Network/Protocol adapters

Network/Protocol adapters have two main functions. They enable different kinds of terminals to access UMF applications and they also provide communications resources to the applications. Each network adapter implements a protocol specific Corba interface. By using the Corba interfaces all of the other components, such as the applications running on the top of framework, can communicate with the adapters. A framework can contain one or more network adapters depending on the environment and the

needs of the service provider. The number of the adapters can also vary over time.

In the diagram below, the SMTP adapter is used to access the UM application. This application uses the SMS adapter to send a short message notification to the mobile phone of the receiver of the email.

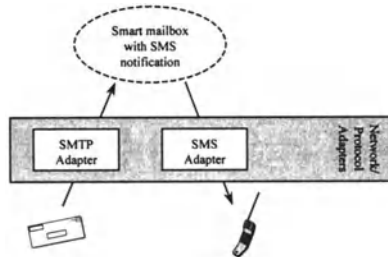


Figure 4. Network/Protocol adapter services

A UM application developer can freely combine the services of the adapters and create any type of sophisticated and advanced services, which may be required. Because new adapters can be easily plugged to the framework, the framework does not limit the imagination of the developer.

The following examples detail possible uses for the network/protocol adapter services. Note that only voice and SMS adapter is presented:

SMS adapter allows its clients to

- send short messages; and
- receive short messages.

Voice adapter allows its clients to

- receive a call;
- make an outbound call;
- detect DTMF tones;
- generate DTMF tones;
- play voice messages; and
- record a voice message.

A voice adapter with the capabilities presented above, enables developers to create, for example, sophisticated voice mail solutions.

4.1.3 Common service components

The main thrust behind the framework approach is that there are common needs among the components of the environment. For example, most of the UM applications need a place for messages and therefore there should be a common message store for the use of all the applications. This kind of general requirement can be solved once as a common UM service

component which provide either Corba interfaces or proprietary interfaces to the client components.

Most of the common services are involved in management since the same management needs apply to all the applications and the UM framework itself. By providing a suitable set of common service components to the use of the UM applications and other components, the framework can be managed as a whole. This way a person who manages an environment like described in this paper has a consistent view to the framework. If these management issues would be solved separately by all of the applications, the result would be unmanageable. Management issues of the UMF have been described in chapter 5.

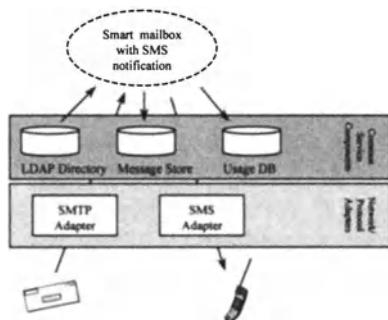


Figure 4. Example of using some common service components

In the example presented in figure 5 the common services used are the LDAP directory for user information, a common message store for the messages and a billing database for accounting. The application, which manages the mailbox can use the directory to retrieve the GSM number of the user who received the email. Since it is possible that other applications also use the same GSM number information, it is stored into a centralised LDAP data store so that it is available for all the UMF components. The retrieved GSM number can then be used to deliver a short message to the receiver of the mail. Because this kind of feature is not free, at least in this scenario, an accounting record is inserted to a common usage database which contains information referring to the usage of the services.



Figure 6. Common components of UMF

As a minimum the following common components are required in the UMF:

- Message conversion utilities for converting messages from one format to another (e.g. text-to-speech).
- A message store as a central repository for the messages.
- A directory, which is used to centralise the configuration information.
- A set of management tools.

Directory and management tools are addressed later in this document.

5. MANAGEMENT

A service framework such as UMF must take care of the management aspects. As mentioned earlier, there are two reasons for this:

- It is not practical to implement the management functionality individually by each UMF component.
- The use of the same management components enforces the whole UM environment into a single management policy.

The components of the framework should form a single Managed Information Base (MIB) for use by the system managers. The MIB must also be easily expandable since the configuration of the UMF will undoubtedly vary over the time.

A traditional approach to address management issues is the use of standard management models such as SNMP or OSI. There are also new ideas based on distributed object based management from the Directory Enabled Networking (DEN) working group and OMG. The DEN approach stores static management information into LDAP directory, but it does not address the management of transient information, which must be managed by the components themselves and accessed by using traditional methods such as SNMP protocol.

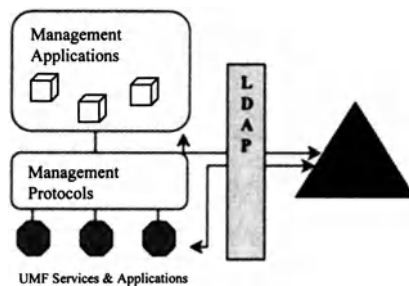


Figure 7. Management overview

Since the main aims are to provide a consistent view for management applications into the UMF MIB and a uniform way to store configuration data, then the DEN proposal offers the best solution. Static management information is stored in the LDAP directory provided by the framework instead of in the configuration files. In this manner a consistent view of the managed information is available without the use of application or component specific configuration files.

The use of LDAP also provides other benefits; the location of the UMF configuration information depends on the directory setup, which allows free placement and replication of this information. Placement and replication can be important issues when considering the performance and availability of the directory. LDAP also solves security and access control issues.

A single view to the transient information of the UMF components can be gained through the use of extensible SNMP agent technologies which enable a single SNMP master agent to be extended when new managed components are added to the framework. Extensibility is needed because SNMP architecture states that one physical machine contains only one SNMP agent which takes care of the management requests.

6. IMPLEMENTING UM FRAMEWORK WITH APPLICATIONS

The creation of an expandable UM framework for executing UM services takes time and requires resources yet, increasing competitive pressures demand that service providers deploy all new services at a fair price and in timely manner. This section provides a concrete roadmap for the development of UM framework, which meets existing and future requirements.

6.1 UM implementation strategy in nutshell

The roadmap for building a reusable framework together with application has been tested in real life. The following steps can be used:

1. Get the UMF model in place. Framework is a goal that the design team takes – no framework is being created accidentally.
2. Define the technical UM solution candidate(s). UM solution comprises the framework together with the UM applications.
3. Implement one of the solution candidates. Pick that solution candidate that best matches organisation's requirements.

6.2 Getting the UMF model in place

Clearly the design of the current interfaces provided by the various existing systems in the service provider environment provides a logical start to the project. The key issue here is to recognise commonly needed resources that will belong to the UM framework. The allocation of responsibilities appropriately among the existing and new software components will result in a robust model that guarantees UM framework expandability and UM service maintainability.

6.2.1 UM application functionality modelling

IT system, such as the UM framework with applications, is typically a black box that will deliver the required functionality after implementation. The following diagram illustrates the well-known email notification example. The UM system boundaries, the participating roles and necessary interactions are modelled. It is to be noted that the ‘email notifier’ function actually delivers short message to SMSC that is responsible for delivery of the message further to the email receiver.

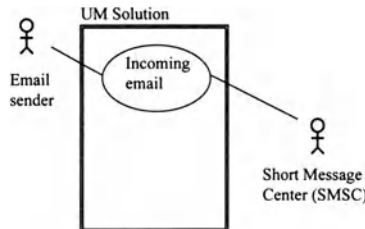


Figure 8. Email notify application requirement modelling

Now the UM application functionality has been modelled. The use case modelling is good in describing the needed functionality. However, no ‘black box’ functionality modelling technique promotes reuse of common components.

6.2.2 UM framework functionality modelling

The modelling technique shown in the previous chapter may result in correct email notification service functionality. However, it does not drive the implementation in the direction where the UM framework is clearly separated from the UM application functionality. To avoid the possible monolithic system implementation, a second modelling phase is needed for

the UM framework part. The following diagram illustrates the functionality being provided by the combination of UM applications and the reusable framework components.

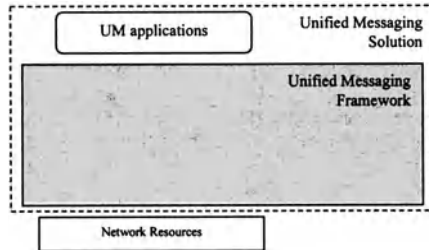


Figure 9. Division into framework and application part

This approach forces the solution to promote a separate interface for the framework part, which is needed for UM functionality expandability and maintainability reasons.

6.3 Define technical solution candidates

Technical solution candidates are being defined to fulfil the model that contains requirements for the UM services and the UM framework.

This section offers two scenarios for development of a new UM service. The first approach is based upon a tool centric start-up, observing the framework implementation from the service perspective, while the second start-up approach is framework component centric.

6.3.1 Tool centric start

Several vendors in the market are already offering UM solutions. These solutions normally provide a complete ‘package’ with set of ready-made UM services. Expansion of this kind of single vendor environment in accordance with the framework requirements described in this paper will enable swifter production of a UM service framework by comparison with the ‘starting-from-scratch’ approach.

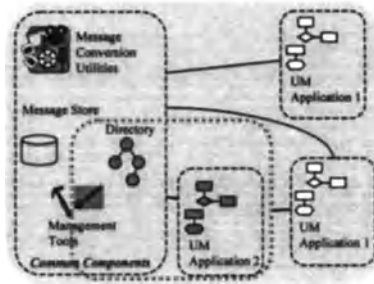


Figure 10. Selecting a UM tool and integrating it to the environment

Selecting a UM solution package requires a commitment to single vendor solution. This approach can involve limitations since telecoms equipment vendors tend to produce closed systems which can be extended only by themselves. Access to the solution's internal resources may limit the opportunities to integrate the tool with external components needed when implementing new services. The challenge for the service provider is to offer the services and service variations for the customers' changing needs while the environment may not provide the full implementation flexibility.

The best possibility would be to find a solution which can be easily tailored and extended by the service provider or a third-party software house.

6.3.2 Framework component centric start

The alternative method for building a framework is to start from the UM service needed and at the same time to produce a piece of the UM framework by abstracting common components from the services to be common framework components. The idea here is to enhance the UM framework with new common components while implementing subsequent UM services.

The standardisation of the management, directory and other components provides an opportunity to drive the implementation to more open direction. However, this approach will require a lot of integration and planning work to combine the UM framework components into a coherent single system, and consequently take longer to start the implementation of the actual UM services.

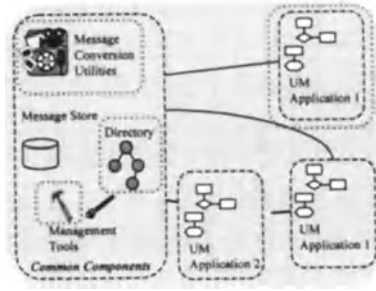


Figure 11. Developing UM framework while creating UM services

6.4 Implementing the solution

Since the core elements of the UM framework, such as the LDAP directory, are already available as commercial software packages, the main thrust of the process is the integration of these applications. Integration is typically implemented by utilising application programming interface (API's) that is a set of application's programmatic services. Because the framework presented here uses a distributed processing platform based on Corba as a communication method between the components, the integration must be implemented with its means. This implies that every component must provide Corba IDL interface for the use of other components.

7. SUMMARY

The unified messaging framework (UMF) approach offers fast time to market for new messaging services through utilisation of existing resources. Reuse of existing resources also is a cost efficient way to develop new services.

For gaining results both in short and long term the UM framework implementation strategy is being proposed in this paper. Efficient and cost-effective UM service development and maintenance require reusability and centralised management which are such important productivity factors that the way of achieving those features is not that important though.

Operators not offering value added services such as UM will have to continue to rely upon price differentiation. If the decision is to offer value added messaging services then the framework based UM solution makes it possible to build and manage messaging services efficiently and cost-effectively.

CONNECTION AND LOCATION MANAGEMENT BASED ON A TINA-COMPLIANT ARCHITECTURE FOR UMTS

Hoang Nguyen-Minh, Harmen R. van As

Vienna University of Technology, Institute of Communication Networks

Key words: TINA (Telecommunication Information Networking Architecture), UMTS (Universal Mobile Telecommunication System), Connection Management, Location Management (LM).

Abstract: Connection and location management are the main processes for a mobile communication network. In this paper, the next generation mobile communication network such as UMTS (Universal Mobile Telecommunication System), is proposed to integrate with a TINA-compliant architecture enabling to handle that kind of management-specific processes. The approach provides a key framework for the unified and adapted provision of connection mobility management toward roaming users using different end systems and, access networks via a core/fixed network. It can be seen as a benchmark application of TINA on UMTS.

1. INTRODUCTION

The current UMTS standardisation activities have been based on a functional type of architecture influenced by Intelligent Networks (IN). UMTS aims to address a scenario with multiple, competitive services and different connectivity providers. In this scenario, IN would take care of user and mobile controls with the related data, while B-ISDN is responsible for the basic switching. It is questionable whether the designated concepts of IN will ever provide the required powerful mechanisms needed for service provisioning, numbering, as well as connection and location management in multi-service provider domains. TINA (Telecommunications Information Networking Architecture) proposes a framework for future telecommunication systems based on distributed and object-oriented

computing that enable fast services provisioning and management, independently of the underlying technology of the networks. TINA offers many advantages over this type of architecture. Applying TINA to UMTS can provide more flexible open system [11,12].

The motivation for studying how UMTS can take advantage of the TINA services and network management concepts is that the IN and TMN standards (the current basis for UMTS) are likely to evolve towards TINA. Here, we argue that a TINA compliant architecture, which already contains some well-developed concepts, can be used to satisfy the requirements of UMTS management such as connection and location management.

The paper is structured as follows: the next two sections present the overview of the UMTS architecture and our proposed TINA-compliant architecture includes a Business Model, Terminal Provider Domain, and Network Resource Information Model. Based on this model, the Network Topology Configuration Management, Connection Management, and Location Management are analysed and implemented.

2. UNIVERSAL MOBILE TELECOMMUNICATION SYSTEM (UMTS)

2.1 UMTS architecture

The basic UMTS network consists of three sub-networks that are radio access network, core network and service network [1,2]. The most complete configuration, the UMTS network architecture, is shown in Figure 1 and its elements are described below.

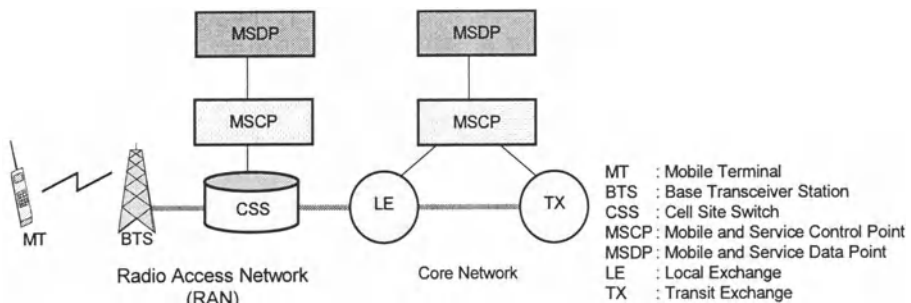


Figure 1. UMTS network architecture

- RAN (Radio Access Network) provides mainly radio-related functions including the basic local switching and transmission functions, measurement reporting, and handover.

- Core Network consists of local and transit switches of the fixed network, including the transmission system, to provide switching, call control, as well as connection control. In UMTS the prescription on the core is to be compliant with the B-ISDN. Within the current definition of B-ISDN, ATM is defined as the transport infrastructure.
- Service Network provides service and mobility control. It also supports call and connection control when the core network allows for the extraction of that functionality from the individual network elements.

2.2 UMTS mobility services

- User Registration: before a user can make outgoing and receive incoming calls, the user must register to a terminal and the terminal must register to the network.
- Location Management: when a user moves to another location area, a location updating procedure must be done to inform the network of the user's current location.
- Handover: the change of physical links (radio channels) and/or fixed terrestrial channels while maintaining a call.

3. THE TINA-COMPLIANT MODEL

3.1 The TINA-compliant business model

TINA-C defines the Business Model [3], which aims at giving an additional structure that eases the application of TINA in a multi-stakeholder, multi-vendor environment. It does so by defining the interaction between business administrative domains in terms of contracts, business roles and business administrative domains. The following entities are to be defined.

- *Consumer* uses various kinds of services provided by a TINA system. A consumer can use services from retailers after subscribing to them.
- *Retailer* provides different kinds of service to customers. In order to a Retailer to provide its services, support from providers playing other roles might be used, e.g. Broker, third party provider and connectivity provider roles.
- *Broker* provides stakeholders information that enables them to find other stakeholders and services in the TINA system.

- *Third party service provider* supports services to retailers or other third party service providers. Unlike a retailer, a third party service provider does not have any contractual relationship with customers.
- *Connectivity provider* owns and manages a network infrastructure which can contribute a transport network to support stream binding in a TINA system or a kTN (kernel Transport Network) supporting a DPE node interconnection.

Their business model then defines how the above roles translate into object domains with precise boundaries and well-defined interfaces, called reference points, between them. The model contains a set of concepts and guidelines to specify reference point, which in turn specify conformance requirements for the TINA system. However, in an environment supporting personal mobility and a multi-provider domain, the relation between user and terminal is not a fixed one-to-one association, but user may register to different terminals for different services, thus allowing a terminal to be shared by different users. Therefore, in the current TINA business model, the consumer role implicitly incorporates a terminal and this modelling does not support all functional of personal mobility because there is no clear separation between terminal and user.

In DOLMEN-AC036 Project [8,9], the new concept of Terminal Provider is defined to solve that problem. It is a stakeholder that owns and manages the terminal equipment information. Its introduction has been recently accepted within the TINA Next Generation Mobile Workshop.

In the context of the third-generation mobile communication network, the connectivity provider owns a set of base stations, an associated access network, and a part of the core network covering a certain geographical region. A retailer may have contracts with several connectivity providers, and each connectivity provider may serve several retailers. Consequently, the coverage area of a retailer may consist of many connectivity provider networks. Figure 2 shows our proposal of some main business roles to model mobility management functionality for the UMTS network by using the DOLMEN-TINA business model. This model includes some additional computational objects (COs) in the computational model in connectivity provider and network topology configuration management domain that can be considered as a TINA-compliant model to implement connection and location management for UMTS.

- *User registration/deregistration* business role allows a user in the *customer domain* to inform a service provider in the *retailer domain* about the terminal from which a user wants to receive or activate services. Deregistration is used to reset an association between user and terminal previously established through registration. These procedures require *access*, *identification*, and *authentication* procedures to be carried out before or in conjunction with them. In

order to be successfully carried out, registration procedures require that the owner of the terminal has allowed access to the terminal. This knowledge can be retrieved from the terminal profile data.

- *User subscription* allows a user (*customer domain*) to subscribe to services provided by the service provider (*retailer domain*). It is split into different subprocedures which include online or offline subscription, changing the previous service setting, dynamic service updating and dynamic subscription of the new-third party service.
- *Location update* allows the network provider (*connectivity provider domain*) to update and store the new location area of the mobile terminal (*terminal provider domain*). Location monitoring operations are started automatically by the mobile terminals on behalf of the users. The concept of terminal location depends on the cellular structure of the area covered by the mobile system. Each mobile terminal continuously receives the location information broadcast by the base stations and compares the received location information with the location information stored in the terminal.
- *Domain update* is similar to location update, but it will be done at the retailer domain. A domain is an administrative area in which a UMTS provider offers a set of UMTS services. Domain update is the part of location monitoring that ensures that the system has up-to-date location knowledge of the terminals and associated users. When the terminal decides to access a new administrative domain, it can execute the domain update procedure.
- *Handover/Paging* are procedures that take place between the terminal provider domain and connectivity provider domain to perform the functionality of terminal mobility. These procedures will be used to support terminal mobility that is out of scope of this paper.

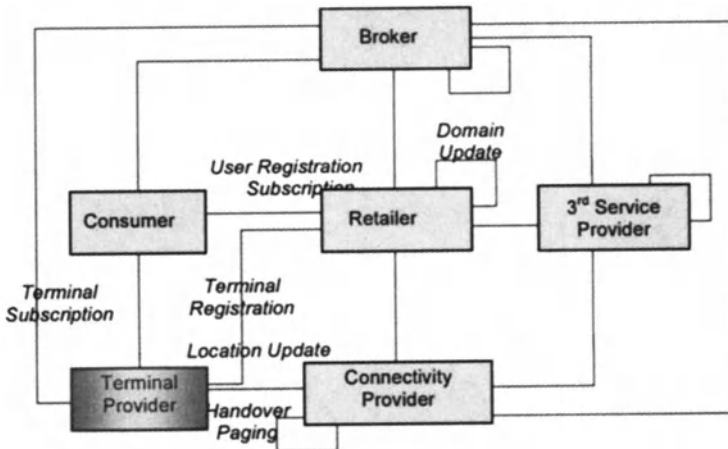


Figure 2. TINA-DOLMEN Business Model for UMTS

3.2 Terminal provider domain

To support personal mobility in a multi-retailer domain environment, the concept of terminal provider has been introduced as that a stakeholder owns and manages the terminal. The reasons for this choice are strongly related to the distinction between user and terminal that characterises personal mobility services. The Terminal Provider requires some new computational objects (COs) representing that role both inside the system and the terminal itself [8]. As can be seen in Figure 3, we suggest an agent associated with the terminal representing the terminal into the system, referred to as a *Terminal Agent* (TA) object. This TA is fixed, even if the terminal is roaming, and should be located in the domain where the owner of the terminal has subscribed. Thus it can always be retrieved through the terminal provider identity with the support of the broker.

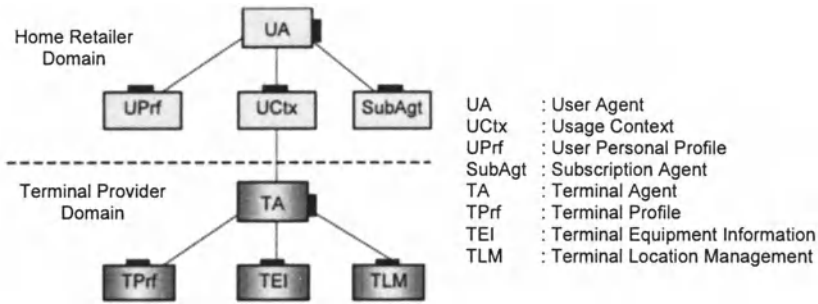


Figure 3. User Agent and Terminal Agent

The dynamic associations between a user and the terminal he has registered on for the different services are maintained in the UCtx (User Context) that points to the TA related with the terminal in the terminal provider. The TA in the Terminal Provider holds the required information to access and use a terminal. In addition to regular consumer information, a terminal provider should maintain the following information: terminal location (in terms of connectivity provider domain), registration policies, and terminal capability. The information needs to be maintained inside the network by the *Terminal Equipment Information* (TEI) in order to support remote registration and subscription procedures. Relating this information to the Terminal Provider makes it possible to locate the terminal without any knowledge of the users registered on it. The *Terminal Personal Profile* (TPrf) manages specific owner preferences and restrictions that can be used to handle the access, identify procedures before the user can use the terminal for receiving and activating services. The information of a list of end-users registered at the terminal could, alternatively, be stored in the terminal itself.

However, it would be more efficient to maintain this information inside the network, thus reducing the traffic on the terminal access link.

Moreover, in order to perform location updating of the mobile terminal when roaming, a new object is also proposed in the terminal, namely Terminal Location Management (TLM). A TLM always listens to the broadcast channel, and each time the TLM recognises a new location area, it initialises a location update by invoking a corresponding operation at the TA.

3.3 Network Resource Information Model (NRIM)

The TINA NRIM [6,13] is a technology-independent network-level resource model that provides a view of the information bearing entities, their relations to each other, as well as the constraints and rules governing their behaviour. In order to master complexity, the model is divided into *fragments*, each defining object classes related to a specific domain that will be described in the following subsections.

3.3.1 Network fragment

A network as viewed by a specific stakeholder who offers connectivity services in Connectivity Provider Domain is called a Connectivity Layer Network (CLNW). A network architecture is an overlay architecture on top of CLNW and is organised by co-operating connectivity providers. The CLNW represents the highest level of abstraction and assembles all resources (e.g. switches, links, adapters, inter-working units and bridges) available to provide network connectivity between access-points of a worldwide transport network.

- *Layer Network (LNW)*: A CLNW is structured in terms of LNWs, with possibly *adapters* between them. The layer network represents all resources involved in the delivery of certain characteristic information in a given format. In the case of a mobile network, some additional control aspects should be supported by a LNW within the mobile environment such as multi-bearer capabilities, macro-diversity, and handover.
- *Adapter* represents a resource that enables a peer-to-peer relation between layer networks that convert the data format and control message used in one layer network to the other.
- *Layer Network Domain (LND)* represents the part of a layer network that is under the control of one administrative domain. From a connectivity provider perspective, a LNW is seen as being made up of one or more LNDs. For the sake of performing and controlling

connection management within a LND, the concepts of *Local Layer Network Domains (LLNDs)* and *Foreign Layer Network Domains (FLNDs)* are introduced. Establishment of a connection crossing different LNDs within the same administrative domain requires co-operation (*federation*) between different LNDs.

- *Subnetwork (SNW)*: A LND consists of a subnetwork (referred to as the top-level subnetwork) on which one or more links terminate. Each such link interconnects the top-level subnetwork of a LND with either the top-level subnetwork of another LND or Customer Premises Equipment (CPE). The top-level subnetwork of a LND may in turn be composed of two or more subnetworks (referred to as lower level subnetworks) interconnected by links. This subnetwork decomposition may occur in multiple levels until the subnetwork maps directly to a network element (NE).
- *Link and Topological Link (TL)* are topological components used for representing an interconnection between a subnetwork and either another subnetwork or a CPE. A Topological Link is a logical or physical transmission path interconnecting two NEs (more generally two subnetworks) and it is supported by a Trail in the server layer network. A topological link has a specific bandwidth capacity that is determined by the bandwidth of the server layer trail. A link can be configured by assigning either a portion of the bandwidth or the entire bandwidth of one or more topological links.

3.3.2 Connectivity fragment

The connectivity fragment describes the information objects and relationships used to model connectivity resources across a TINA network.

- *Network Flow Connection (NFC)* models the end-to-end connection between CPEs across a CLNW.
- *Terminal Flow Connections (TFC)* models the terminal part of Stream Flow Connection.
- *Trail* models an end-to-end connection within a layer network.
- *Subnetwork Connection (SNC)* models a switched connectivity across a sub-network (point-to-point or involving multi-point topologies).
- *Link Connection (LC)* is a connection between either subnetworks or between a sub-network and a CPE.
- *Tandem Connection (TC)* is a chain of subnetworks and link connections.
- *Edge* is a leg of a subnetwork connection. Root and branch relations to model multi-point connectivity.

3.3.3 Termination Point Fragment

Termination end points in a CLNW are modelled in a separate fragment. This fragment includes Network Flow End Point (NFEP), Trail Termination Point (TTP), Network Trail Termination Point (NWTTP), Network Connection Termination Point (NWCTP), Link Termination Point (LTP), and Topological Link Termination Point (TLTP). These concepts are used to model end points between CPEs across a CLNW, trails, subnetwork connections, links and topological links, respectively. The Stream Flow End Point (SFEP) is also defined, to model the sources and sinks of the information flows.

A topological point on the boundary of a connectivity layer network (i.e., resident in a CPE) that represents the potential for several network flow terminations to occur at the point is called a Network Flow Endpoint Pool (NFEPPool). A NFEPPool is a collection of LTPs, and the LTPs grouped under a NFEPPool may span different layer network domains of the same connectivity provider.

3.3.4 NRM of UMTS network

The complete NRM for the UMTS system is shown in Figure 4. The Mobile Terminal is modelled as a CPE object. This object has been introduced so that end-to-end connectivity, stream flow connection (SFC) and stream bindings can be represented in the view of a connectivity provider. It is not suggested that a connectivity provider is aware of all CPEs included in the UMTS network. However, when the mobile terminal roams to the visited connectivity provider domain, it triggers a procedure to update the new location to the current connectivity provider and inform the position change to the home domain. Such foreign CPEs will dynamically appear and disappear (i.e., they are transient objects) in the view of the connectivity provider. In accordance with [7], CPE object contains SFEPs, which are mapped dynamically onto NFEPs in the CNLW by TFCs. The RANs and Core Networks are modelled as layer network or subnetwork objects independently of the technology used in the real networks. The features of macro-diversity and multi-bearer of UMTS will be supported by adding the control aspects of LNW and SNW. Details of NRM for UMTS that completely supports functionality of multi-bearer and macro-diversity can be found in [10,13].

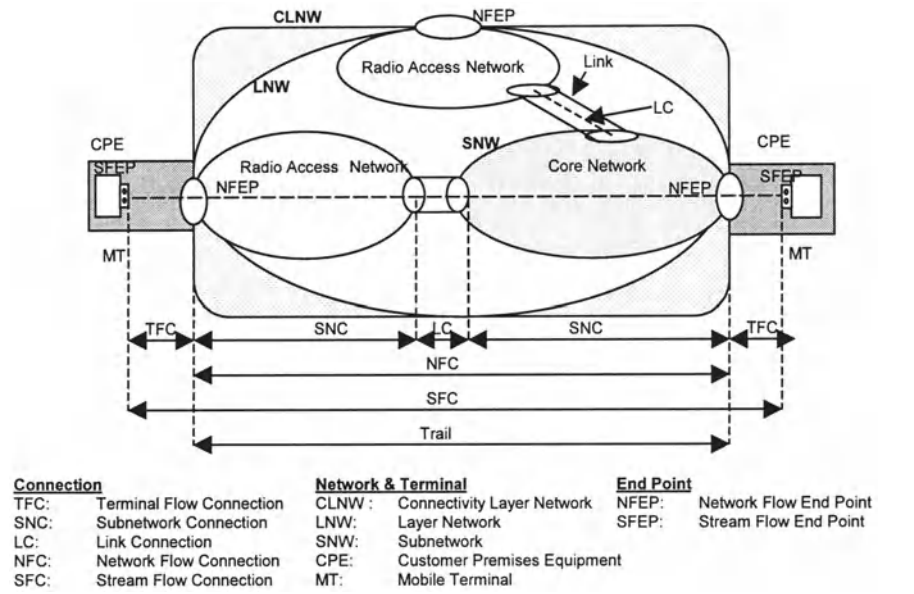


Figure 4. NRIM Model of UMTS Network

4. NETWORK TOPOLOGY CONFIGURATION AND CONNECTION MANAGEMENT

According to TINA [4,5], Network Configuration Management consists of Network Topology Configuration Management (NTCM), which deals with *static network resources*, and Connection Management, which deals with *dynamic network resources*. The following subsections describe these in detail.

4.1 Connection Management

Connection management is the activity of co-ordinating allocation and release of the telecommunication services (distributed applications) of connectivity network resources on behalf of a manager. Connection Management functions may also support other management services (for instance, a fault management application that needs a connection through a specific trunk in order to test that trunk), and the interconnection of DPE nodes. The Connection Management Architecture is based on the TINA Network Computational Model that is divided into a number of conceptual levels such as communication, connectivity, layer network, and subnetwork level (Fig. 5). This grouping is also used to explain the functionality of each object and its relationship with other objects. The proper behaviours of COs

are added in order to minimise the set of operations (with parameters) and coherently setup and release connections.

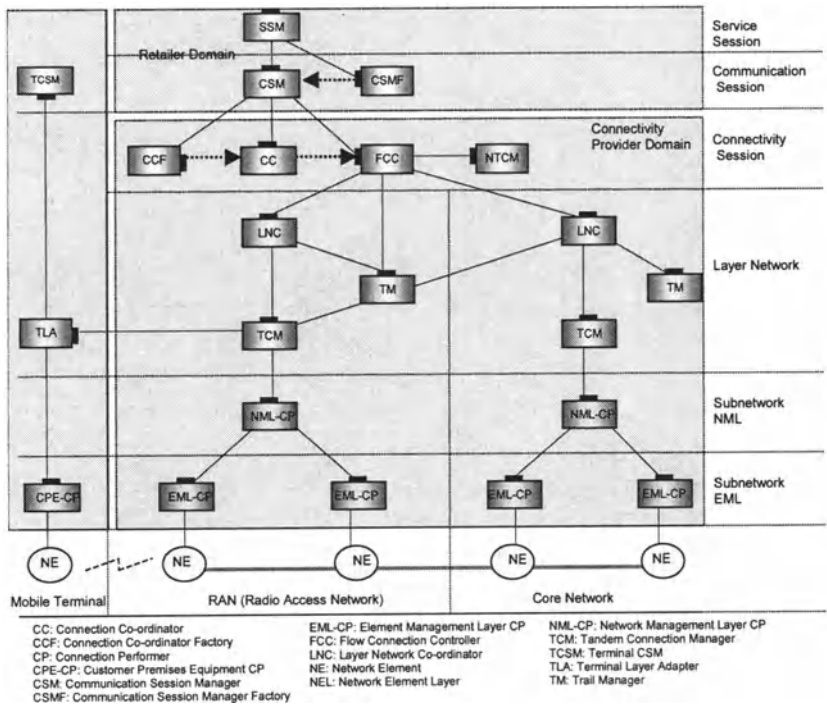


Figure 5. Computational model for Connection Management

Whenever a communication session is required, the CSM (Communication Session Manager) should be contacted in the retailer domain. As a result, a stream binding is setup at the service level and a set of Stream Flow Connections (SFCs) is established in order to accomplish the service session demands. The CSM uses the Stream Flow End Point (SFEP) descriptions to determine the associated Terminal CSM (TCSM) to negotiate with the terminal involved the information needed to perform that communication session. The CSM contacts each TCSM associated with the SFC so that the terminal and network parts of the SFC can be connected. The CSM translates the SFCs to Network Flow Connections (NFCs) and uses the connectivity session to establish these connections.

The connectivity session presents a technology-independent network view that is independent of complications such as multiple connectivity provider domains or establishing connections over different network technologies. This view is supported by the NFCs, which are modelled by the Physical Connection Graph (PCG). However, to establish connections, this abstract view must be translated to a suitable technology-dependent

layer network view. The connectivity session is responsible for connecting NFEPs and realises this by partitioning the NFC into *trails* and determining the layer networks that can support the characteristics of these trails. A total of three COs reside in the connectivity session. They are CCF (Connection Co-ordinator Factory), which serves as a factory object for connectivity session; CC (Connection Co-ordinator) which acts as session control; and FCC (Flow Connection Controller) which acts as connection control. An important selection criterion for choosing a certain layer network is whether the layer network can support the QoS requirements as captured in the description of the NFC.

The LNC (Layer Network Coordinator) and subordinate objects (NML-CP and EML-CP) present an end-to-end view of a network technology, and offers services to establish *trails* across the entire network. Upon connectivity session setup, the FCC contacts the NRCM (Network Topology configuration Management) domain to determine the layer networks that can support the characteristics of the required trails. The layer networks are managed and controlled by LNCs, such that the FCC requests LNCs to set up the trail. The LNC consequently creates a TM (Trail Manager) that is held responsible for providing manipulations on the trail. A TM instance exists for each trail within a layer network and its reference is forwarded to the FCC such that this object can invoke actions. To a peer layer concept (components acting on behalf of other domains of the same layer network), the LNC offers services to setup *tandem connections* to complete trails that are terminated outside their domain. A CO called TCM (Tandem Connection Manager) is created by TM to manage the above tandem connections.

The side of the layer network that resides in the MT (mobile terminal) is managed and controlled by the TLA (Terminal Layer Adapter). This computational object provides functions for creating, manipulating and deleting NFEPs. The CPE stores one TLA for each network technology it supports. This object is created upon configuration of the terminal and has a maximum lifetime coinciding with the lifetime of the terminal. Upgrading a terminal may result in changing the TLA. The TM and TLA exchange details of the NFEPs that will be used for the network connection.

4.2 Network Topology Configuration Management

The Network Topology Configuration Management (NTCM) [4] introduces some aspects of connectivity resource configuration (connection management) in order to help understanding the distinction of topology resource configuration management and connectivity resource configuration management. The main goal of NTCM is to administrate the life cycle of the installed topology resources (create, delete, update and status change of the

resources). The functions that should be provided by the NTCM can be defined as follows:

- Maintaining the resource map of the network resources, in the viewpoints of inventory of all resources and network topology (the relationships between resources).
- Allowing activation, deactivation, reservation and release of resources.
- Installation support that can support the installation and removal of network resources.

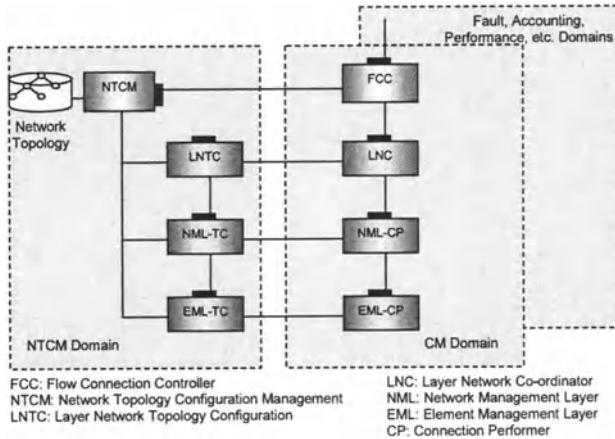


Figure 6. Computational model of Network Topology Configuration and Connection Management

Routing function that can support the procedure for routing information exchange between LNDs and connectivity provider domains. Routing information can be used (by FCC) to choose a route within the layer network or federate with the neighbouring domains to create tandem connections.

The Network Topology Configuration Management is modelled as a computational object (CO) named as NTCM (Network Topology Configuration Manager) as described in Figure 6. The interfaces among NTCM and other management domains (Fault, Configuration, Accounting, Performance and Security – FCAPS Management) are considered and COs within NTCM are defined. For the scope of this paper, we consider only the Connection Management that addresses the management of *dynamically* allocated connectivity resources. Other Subordinate COs are defined in the NTCM domain as follows:

- Layer Network Topology Configuration (LNTC)
- Network Management Layer Topology Configuration (NML-TC)
- Element Management Layer Topology Configuration (EML-TC)

They configure the topology resources of the layer network, network management layer and element management layer levels respectively. The

Network Topology database is an inventory of network resources and the relationship between them that should be maintained by NTCM.

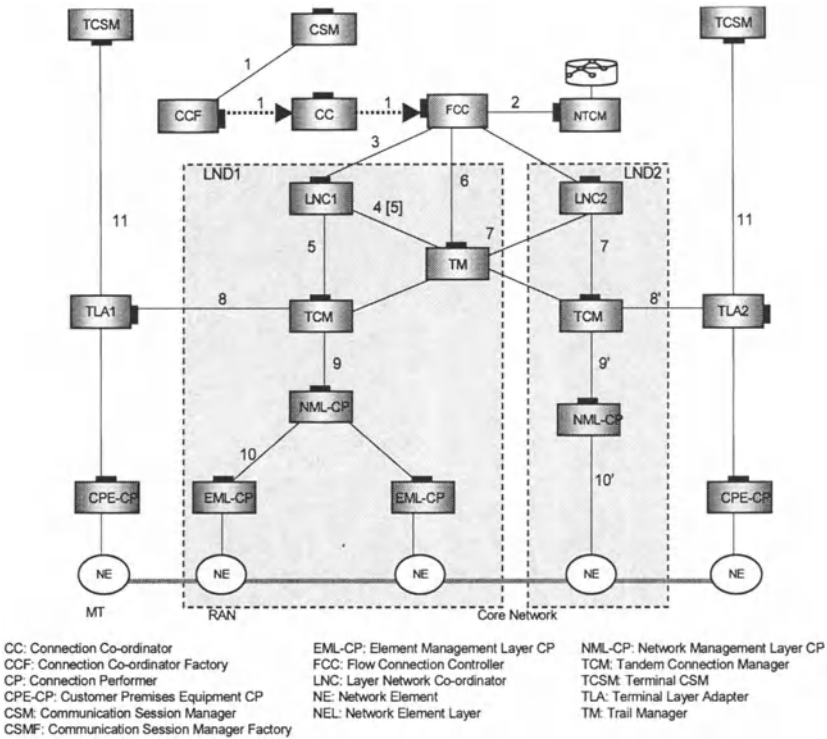


Figure 7. Connection management scenario

5. CONNECTIVITY SESSION SETUP SCENARIO

This section describes the procedure that is necessary to setup a connection in a UMTS system, between a mobile terminal and another terminal, either mobile or fixed. In order to make clear the control of the connections a simple network configuration will be considered where only one connectivity provider domain with two LNDs are involved. In this scenario, a computational diagram (Figure 7) is provided where the interactions between the different COs involved are shown. This representation is more focused on high-level mechanisms rather than on technology specific ones. In addition, the LND federation relationship is not further described here due to its complexity. Details concerning this issue can be found in [15].

1. CSM requests the CCF to setup a new connectivity session; CCF creates a new CC for that connectivity session; then for each NFC in the connectivity session CC creates a FCC to manage it.
2. FCC requests NTCM for the current information of network topology to make a decision which is initiated layer network domain for the trail associated with that NFC (FCC translates NFC into NWTTPs of the trail).
3. FCC locates the LNC1 (initiated layer network object described in step 2) of LND1 and requests a trail setup.
4. LNC creates a TM to manage the new trail; then LNC requests TM to add branches to the new trail.
5. To attend this request, TM requests the creation of a tandem connection within LND1 to LNC1 and LNC1 creates a TCM1 for the tandem connection.
6. As the above assumption, one of NWTTPs may in this scenario not belong to the LND1 that TM should request FCC to solve which LND is involved in the trail. This is performed by the federation between FCCs in multidomains. Then the reference of LNC2 is solved and returned to TM.
7. TM requests LNC2 to add a tandem connection for the trail; LNC2 creates TCM2 for that additional tandem connection.
8. (8') TCM requests the setup of NFEPs from the TLA in the consumer's domain, including the negotiation of connection parameters such as QoS and the bandwidth that can be supported by the terminal to that connection.
9. (9') TCM then requests to its top NML-CP object to create a subnetwork connection among the identified NWTTPs. Each NML-CP deals with routing functionality by interacting with the respective NML-TP in the NTCM domain.
10. (10') Each NML-CP determines the lower CP objects (finally, at EML-CP) to request the connectivity for the subnetworks in the domain. EML-CP objects keep control of the use and reservation of the resources associated with a certain subnetwork (i.e. termination points and bandwidth associated to the subnetwork).
11. TLA selects and requests the corresponding NFEPs and the association of TFC from TCSM; TCSM performs the association request. It is assumed that the nodal binding has been established and the internal interactions of the terminal are not described here

6. LOCATION MANAGEMENT

This scenario, as seen in Figure 8, is used to monitor, detect and update the location when the mobile terminal changes from the current location area (LA) to the new one. The concept of LA depends on the cellular structure area covered by the mobile system that can be identified by its Location Area Identifier (LAI). It is broadcast to all associated cells by a Location Management (LM) computational object in the connectivity provider domain. That procedure is triggered whenever a terminal moves from one LA to another and forces the TA to update the location information of the terminal stored in the TEI. That information will be used by NTCM to solve the location of the terminal when FCC requests to make a connection with. Depending on the network configuration and behaviour of the mobility user, the terminal location information will be maintained and stored in the Terminal Provider or in the NTCM domain. This issue is opened for further study.

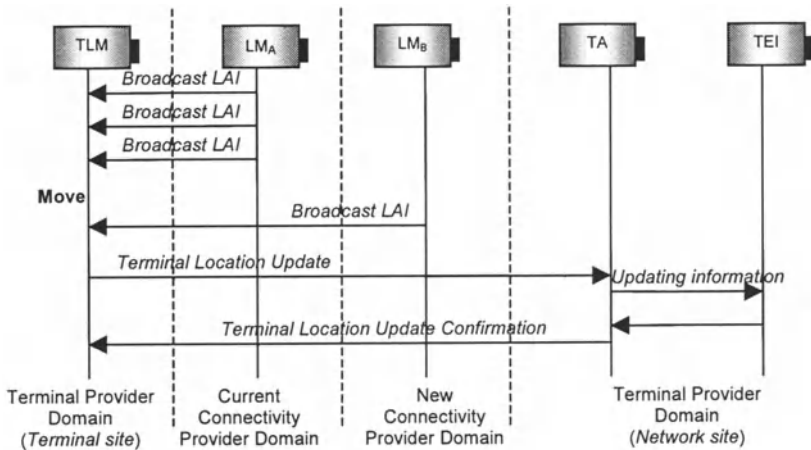


Figure 8. Location Management

7. CONCLUSION

In this paper, we present a TINA compliant architecture as an interesting candidate to perform connection, network topology configuration and location management for UMTS. The proposed TINA-compliant architecture (comprising of Business Model, NRIM and Computational Model) is suitable for model the UMTS network to handle diverse kinds of UMTS requirements. The Business Model defines the main stakeholder roles

and clarifies their relationships to each other, as well as the extension of the new role of Terminal Provider that can support personal mobility and location management. The network topology configuration management with the proposed routing functionality can be used for fast setup and managing a connection. Further work will concentrate on how to maintain the stream and operational bindings during handover and how to realise a virtual home environment (VHE) to fulfil entirely mobility management supporting UMTS.

ACKNOWLEDGEMENT

This research work has been supported by OAD (Oesterreichischer Akademischer Austauschdienst).

REFERENCES

- [1] UMTS Forum, "The path towards UMTS-Technologies for the Information Society", Report No. 2, 1998.
- [2] W. van den Broek, P. Massafra, "UMTS requirements on IN Call Modeling", RACE Mobile Telecom Submit, Cascais, Portugal, Vol. 2, Nov. 1995, pp. 344-349
- [3] TINA-C, "TINA Business Model and Reference Points 4.0", May 1997.
- [4] TINA-C, "TINA Network Resource Architecture 3.0", Feb. 1997
- [5] TINA-C, "Connection Management Architecture", March 1995
- [6] TINA-C, "Network Resource Information Model Specification", Dec. 1997
- [7] N. Natarajan, "TINA Network Resource Information Model", Journal of Network and Systems Management, Vol.6 No. 3, 1998, pp. 283-311.
- [8] AC036 DOLMEN, "Open Service Architecture for Mobile and Fixed Environments (OSAM) Final Release", DOLMEN/ AS-DEL04, April 1998.
- [9] S. Trigila, K. Raatikainen, B. Wind, P. Reynolds, "Mobility in Long-term Service Architectures and Distributed Platforms", IEEE Personal Communications, Vol.5 No.4 Aug. 1998, pp. 44-55.
- [10] EURESCOM Project P608 Documents, "TINA Concepts for third Generation Mobile Systems", Deliverable 1, 2 and 3, 1997 & 1998.
- [11] EURESCOM Project P847-GI, "TelCo Assessment of TINA", Deliverable 2, Vol.1, Feb. 1999.
- [12] A. Küpper, F. Imhoff, "TINA-compliant Service Provision and Numbering in UMTS", Proceedings of IEE 6th Conference on Telecommunications, Edinburgh, March 1998.
- [13] M. Roberto, P. Flavio, "TINA-based Network Resource Information Model for next-generation Mobile Systems", Proceedings of the TINA'97 Conference, Santiago, Chile, 1997, pp. 308-322.
- [14] P. Juan, et al., "The VITAL Network Resource Architecture", Proceedings of the TINA'97 Conference, Santiago, Chile, 1997, pp. 130-138.
- [15] P. Juan, et al., "Network Provider Domain Federation in TINA", Proceedings of the TINA'97 Conference, Santiago, Chile, 1997, pp. 334-342.

RELATIONSHIP BETWEEN SDL PROCESSES AND MESSAGE SEQUENCE CHARTS

Jie Lian and Vilas Wuwongse

Computer Science and Information Management Program

School of Advanced Technologies

Asian Institute of Technology

Key words: Extended Finite State Machine, MSC, Regular Expression, SDL

Abstract: Specification and Description Language (SDL) and Message Sequence Chart (MSC), two widely used system engineering modeling languages, specify and describe different aspects of real time systems. Since MSC is always combined with SDL in system specifications and descriptions, their interrelationship is very important and deserves special attention. Two new mathematical models for precise description of SDL Processes and MSC are used to analyze their interrelationship.

1. INTRODUCTION

During the last two decades, telecommunication has employed some of the world's largest and most complex computer-based real time systems, which operate in real time with time constraint [1] and put heavy demands on system engineers. Therefore one major concern is the management of complexities, for which SDL [4, 5] and MSC [6, 7] are widely used.

SDL has been developed for two purposes of a precise specification/description of functional properties and the realization of required systems. A SDL system consists of two basic structures: Block and Process, describing system physical components and behaviors, respectively.

MSC, a popular graphical language for the visualization of selected system runs (traces) within a communication system, can also be viewed as a

special tracing language which mainly concentrates on message interchange by communicating entities and their environment [11]. MSC, combined with SDL or any other language, is now used at nearly all stages of the system development process.

Since SDL Processes and MSC can be used to describe system behaviors, they should be interrelated. A detailed analysis of this relationship will not only provide a more precise and powerful interpretation of system specifications and descriptions, but also overcome disadvantages, inherent in these models.

An analysis of this relationship demands two mathematical models. By tradition, SDL Processes are defined by Extended Finite State Machines (EFSM), which are tightly linked to Regular Expressions, whence, to start with, this paper constructs an EFSM and analyzes its relationship to Regular Expressions. Next, MSC is mathematically defined and its relationship to Regular Expressions analyzed. Finally, a relationship between SDL Processes and MSC is obtained. Detailed conversions between these two models are given in the other two related papers [8, 9].

2. AN EFSM INTERPRETATION OF SDL PROCESSES

SDL structures have the two levels of System and Block. The description at the System Level tells the designer that the system and environment interact through signal channels and that each channel is served independently by a block, that at the Block level specifies the components and the signal channels between them.

A Block may contain many Processes which describe the behavior of the system components in SDL. Since SDL Processes are defined by EFSM, their behavior is strictly sequential and concurrent behavior can only occur between Processes.

A Process yields the main dynamic description of system behavior and performs a very important role in SDL systems, since it is defined and formalized by an EFSM. The Finite State Machine (FSM) is a well-known model in computer science, which has been extended to EFSM in SDL, in order to handle inputs with data, outputs as well as tasks of systems.

Nevertheless, there do not exist a formal definition and properties of EFSM, which are used to define SDL Processes. A new version of EFSM is presented below and its properties are discussed.

2.1 The Deterministic Extended Finite State Machine (DEFSM)

An EFSM comprises a set of *states*, a set of *state transitions* which occur once an input has been chosen from an *input* set, and a set of *output and task functions*. The *initial state*, usually denoted by q_0 , starts the machine. In a DEFSM, each *input* has exactly one *transition* out of each *state*, during which there is exactly one *output-task sequence* executed before the *state* changes; its textual definition employs mathematics and automata theory. The theorems, which are necessary for the construction of a DEFSM, follow.

Definition:

If M is a set of DEFSM, then the machine $m \in M$ is the seven-tuple

$m = (Q_m, \Sigma_m, \Theta_m, \Psi_m, \Gamma_{m-ST}, \Gamma_{m-OT}, q_{m-0})$, where

- **Q_m is the set of n states** $\{q_{m-1}, q_{m-2}, \dots, q_{m-n}\}$. According to SDL usage, there exist the three *state* types: Normal, Decision and Initial. Normal State is normal states in an FMS. Decision State refers to the Decisions in an SDL Process, while Initial is a special case of the Normal State which need not receive a signal, but changes automatically to the next *state* and performs certain initializing *tasks*.
- **Σ_m is the set of inputs** $\{i_{m-1}, i_{m-2}, \dots\}$. In_{m-k} is used to denote an *input* sequence $(i_{m-1} \bullet i_{m-2} \bullet \dots \bullet i_{m-p})$, where k is the input sequence number, p the length of In_{m-k} and ' $\bullet$ ' is the sequential symbol. In SDL, there are two *input* types: Normal and Special, the former denoting the signal received from the Environment, while the latter may be TRUE or FALSE or have certain forms, which are only used in the Decision State. In fact, there do not exist TRUE and FALSE signals, emitted from the Environment; the Special Inputs are introduced, in order to construct a unified definition for the SDL Processes.
- **Θ_m is the set of outputs** $\{o_{m-1}, o_{m-2}, \dots\}$.
- **Ψ_m is the set of tasks** $\{t_{m-1}, t_{m-2}, \dots\}$. With the exception of Normal Task, Procedure Tasks denote procedures in SDL, which can be replaced by any EFSM with one entrance and at least one termination point. In certain special cases, it will be useful to denote the Output-Task sequence by a single graphical symbol. A dashed box will be used to denote an Output-Task sequence (Fig. 2.4).
- **$\Gamma_{m-ST}: Q_m \times \Sigma_m \rightarrow Q_m$ is a transition function.** Let the current state be $q_{m-k} \in Q_m$; after a new input $i_{m-p} \in \Sigma_m$ has been received, the EFSM will change to the next state $q_{m-k}' = \Gamma_{m-ST}(q_{m-k}, i_{m-p})$, where ST indicates that it is a State Transition.
- **$\Gamma_{m-OT}: Q_m \times \Sigma_m \rightarrow (\Theta_m \cup \Psi_m)^*$ is an Output-Task (OT) sequence function.** For each transition, defined in a transition set, let the current state be $q_{m-k} \in Q_m$; after a new input $i_{m-p} \in \Sigma_m$ has been received, the

machine will change to the next state and complete an Output-Task sequence of the form $OT_{m-k}=I_{m-OT}(q_{m-k}, i_{m-p})$, where the following condition must be true:

$$(OT_{m-k}=u_{m-1} \bullet u_{m-2} \bullet \dots \bullet u_{m-q}) \wedge (u_{m-1}, u_{m-2}, \dots, u_{m-q} \in (\Theta_m \cup \Psi_m)) \wedge (\exists q_{m-k}': q_{m-k}'=I_{m-ST}(q_{m-k}, i_{m-p})).$$

$IOT_{m-j} = (i_{m-p} \bullet OT_{m-k})$ denotes an Input-Output-Task sequence for the above transition and related OT function. Thus, on receiving an input i_{m-p} , there exists for each transition an OT and an IOT . If OT is empty, then $IOT=i_{m-p}$.

- q_{m-0} is an initial state, an entrance to the EFSM m . Its graphical symbol has been shown in the State Set definition.

Extension of a Transition

For a reception of an input sequence, a transition can be extended

$$I_{m-ST}: Q_m \times \Sigma_m \rightarrow Q_m \text{ to } I_{m-ST}: Q_m \times \Sigma_m^* \rightarrow Q_m, \text{ i.e.,}$$

1. $q_{m-k} = \Gamma_{m-ST}(q_{m-k}, \epsilon)$,
 2. $\Gamma_{m-ST}(q_{m-k}, In_{m-p} \bullet i_{m-n}) = \Gamma_{m-ST}(\Gamma_{m-ST}(q_{m-k}, In_{m-p}), i_{m-n})$,
- where (1) states that EFSM can not change state without reading an input and (2) how to find the state after reading a non-empty input sequence $In_{m-p} \bullet i_{m-n}$.

In EFSM models, there are three kinds of deterministic concepts: Determinism in transition, in the OT function and in both, respectively.

1. An EFSM is called **deterministic in Transition** (abbreviated DEFSM-Tran) if it will only admit zero or one transition from a state at the same input.
2. An EFSM is called **Deterministic in OT function**, (i.e., DEFSM in OT function, or DEFSM-OT), if it only admits zero or one OT from one state to another at the same input.
3. An EFSM is said to be a **fully deterministic EFSM** or DEFSM in full, if it is deterministic in Transition and OT function.

2.2 Non-deterministic EFSM Model

Let $Pow(Q_m)$ denote the power set of set Q_m . If the definition of transition is changed to $I_{m-ST}: Q_m \times \Sigma_m \rightarrow Pow(Q_m)$, i.e., if an EFSM allows zero, one or more transitions from a state to others at the same input, the EFSM will be called Non-deterministic in transition and abbreviated NEFSM-Tran.

If the definition of OT function is changed to $I_{m-ST}: Q_m \times \Sigma_m \times Q_m \rightarrow \{(\Theta_m \cup \Psi_m)^*, \dots, (\Theta_m \cup \Psi_m)^*\}$, i.e., if the EFSM allows zero, one or more OT from a state to another at the same input, the EFSM will be called Non-deterministic in OT, abbreviated NEFSM-OT.

2.3 Basic Types of EFSM

In the sequel, the following abbreviations are used to denote basic types of EFSM:

- DEFSM is deterministic in its transition and OT functions, i.e., it is fully deterministic;
- NEFSM-OT is deterministic in transition and non-deterministic in the OT function.
- NEFSM-Tran is non-deterministic in the transition and deterministic in OT function.
- NEFSM-Full is non-deterministic in transition and OT functions.
- NEFSM is used to denote above three kinds of Non-deterministic EFSM.

All the SDL Processes are denoted by DEFSM-full. The reason for discussing NEFSM is when we analyze relationship between SDL Processes and MSCs, some of MSCs present non-deterministic characteristic which causes difficulty of converting MSCs to SDL Processes. So, the NEFSM is defined to give a unified relationship between them.

2.4 Languages Accepted by EFSM

An input sequence In_{m-p} is said to be accepted by an EFSM, if there exist states q_{m-k} , and $q_{m-k} = I_{m-ST}(q_{m-0}, In_{m-p})$, i.e., if there exists in the transition diagram a path, labeled by the *input sequence* from an *initial state* to certain states, then the *input sequence* is accepted by the EFSM.

The above notion of acceptance can be further extended such that an EFSM may be considered to accept not only *input*, but also Input and Output-Task Sequences. In the EFSM model with OT sequences, the new concept of "Behavior Sequence" (BS) is defined, in order to describe the entire behaviors of EFSM. If m is an EFSM and In_m an arbitrary *input sequence* of the form $(i_{m-0} \bullet i_{m-1} \bullet \dots \bullet i_{m-p})$ accepted by m , the Behavior Sequence BS for In_{m-k} is defined by:

$$BS = IOT_1 \bullet IOT_2 \bullet \dots \bullet IOT_p = (i_{m-1} \bullet OT_{m-1}) \bullet (i_{m-2} \bullet OT_{m-2}) \bullet \dots \bullet (i_{m-p} \bullet OT_{m-p}),$$

where $\forall 1 \leq n \leq p$,

$$IOT_{m-n} = i_{m-n} \bullet OT_{m-n},$$

$$OT_{m-n} = I_{m-OT}(I_{m-ST}(q_{m-0}, i_{m-0} \bullet i_{m-1} \bullet \dots \bullet i_{m-(n-1)}), i_{m-n}, I_{m-ST}(q_{m-0}, i_{m-0} \bullet i_{m-1} \bullet \dots \bullet i_{m-n})),$$

and, if $n=0$, then $q_{m-0} = I_{m-ST}(q_{m-0}, \varepsilon)$ and $OT_{m-n} = \varepsilon$.

IOT is called an Input-Output-Task sequence within one transition. Due to the fact that each Behavior Sequence consists of zero, one or many *IOTs*, and each *IOT* cannot be distributed over one transition function and the relative OT function, it is also called an Axiom Behavior Sequence.

The concept of “language”, accepted by FSM, is like that in automata theory; the set of all behavior sequences, accepted by an EFSM m_k , is called a language accepted by the EFSM and denoted by $L(m_k)$.

For DEFSM, there exists exactly one behavior sequence for each input sequence, while for NEFSM, there may exist one or several behavior sequences. Let $L(m)$ denote the set of all possible behavior sequences, produced by an EFSM m . Each element in $L(m)$ is denoted by BS_{m-k} in the form $IOT_{m-1} \cdot IOT_{m-2} \cdot \dots \cdot IOT_{m-n}$, where k is an integer.

3. MATHEMATICAL DEFINITION OF MSC

Consider the signal exchanges among many Instances of MSC in a system and executions of certain, corresponding internal sessions. Comparing MSC'96 with MSC'92, it becomes obvious that many new language concepts have been added, including Generalized Ordering, Inline Operator Expression, Reference, High Level MSC (HMSC). In the sequel, HMSC has been ignored because it describes the top-level system structure which can not describe by SDL Processes. Meanwhile, Generalized Ordering has been ignored since in this paper only the similar function units of SDL Processes and MSC will be considered.

3.1 Symbols used for the mathematical definition of MSC

Input i : $in(input-name, source)$, i.e.,
 $in(i, env)$ means that Input i comes from the Environment.

Output j : $out(output-name, destination)$, i.e.,
 $out(j, env)$ means Output j is sent to Environment.

Action k : $act(k)$, i.e.,
 $act(k)$ means execution of the Action k .

Reference l : $ref(l)$, i.e.,
 $ref(l)$ is the MSC Reference l .

#: is an ordering symbol with the following three forms:

+: is an alternative symbol. $u_1 + u_2$ means only one of u_1 and u_2 can be chosen, where u_1 and u_2 are one of the basic elements of MSC.

•: is a sequential order symbol. $u_1 \bullet u_2$ means that u_1 is followed by u_2 .

$||$: is a parallel order symbol. $u_1 || u_2$ means that u_1 and u_2 are in parallel.

The priority of the three symbols is $|| > \bullet > +$. So $u_1 || u_2 = u_1 \bullet u_2 + u_2 \bullet u_1$, and $u_1 \bullet u_2 || u_3 = u_1 \bullet (u_2 || u_3) = u_1 \bullet u_2 \bullet u_3 + u_1 \bullet u_3 \bullet u_2$. The parentheses ‘(’ and ‘)’ can be used to indicate the priority of the sub-sequences with the highest priority.

$\{ \dots \}$: beginning and end symbols of a sequence.

3.2 MSC Definition

A single Instance in MSC is a seven-tuple

$$Ins = (\Sigma_{Ins}, \Theta_{Ins}, \Psi_{Ins}, Con_{Ins}, Exp_{Ins}, Ref_{Ins}, \Pi_{Ins}),$$

where

1. Σ_{Ins} is a set of Inputs $\{i_{Ins-1}, i_{Ins-2}, \dots, i_{Ins-n}\}$, if the Instance has n Inputs,
2. Θ_{Ins} is a set of Outputs $\{o_{Ins-1}, o_{Ins-2}, \dots, o_{Ins-n}\}$, if the Instance contains n Outputs,
3. Ψ_{Ins} is a set of Actions $\{a_{Ins-1}, a_{Ins-2}, \dots, a_{Ins-n}\}$, if there are n Actions,
4. Con_{Ins} is a set of Conditions $\{con_{Ins-1}, con_{Ins-2}, \dots, con_{Ins-n}\}$, if there are n Conditions.
5. Exp_{Ins} is a set of Inline Operator Expressions $\{exp_{Ins-1}, exp_{Ins-2}, \dots, exp_{Ins-n}\}$, if there are n Inline Operator Expressions and each exp has the form

$$exp_{Ins-i} (Expression\ Keyword, [BS-List], [BS-List] \dots [BS\ List]),$$

where *Expression Keywords* are “alt”, “par”, “loop”, “opt” and “exc” and *BS-List* is a Behavior Sequence.

6. Ref_{Ins} is a Reference set $\{ref_{Ins-1}, ref_{Ins-2}, \dots, ref_{Ins-n}\}$, if there are n References.
7. Π_{Ins} is an ordered behavior sequence $\Pi_{Ins} = (\Sigma_{Ins} \cup \Theta_{Ins} \cup \Psi_{Ins} \cup Con_{Ins} \cup Exp_{Ins})^*$, where each element in $(\Sigma_{Ins} \cup \Theta_{Ins} \cup \Psi_{Ins} \cup Con_{Ins} \cup Exp_{Ins})^*$ is connected by ordering symbols.

After dealing with all Instances of MSC, an MSC is defined by the two-tuple:

$$Msc = (I, F),$$

where

1. I is the set of instances $\{Ins_1, Ins_2, \dots, Ins_n\}$.
2. F is the ordering function of the MSC,

$$F: \underline{I}_{Ins1} \times \underline{I}_{Ins2} \times \dots \times \underline{I}_{Ins_n} \rightarrow \lambda(\underline{I}_{Ins1} \parallel \underline{I}_{Ins2} \parallel \dots \parallel \underline{I}_{Ins_n}),$$

where the symbol λ is the *state operator* used to enforce the basic static requirement that a message must be sent before it is received [10].

The following examples explain these definitions further and the graphical symbols of recommendation Z.120 (MSC'96) will be used below.

3.2.1 MSC of One Instance with Sequential Inputs and Outputs

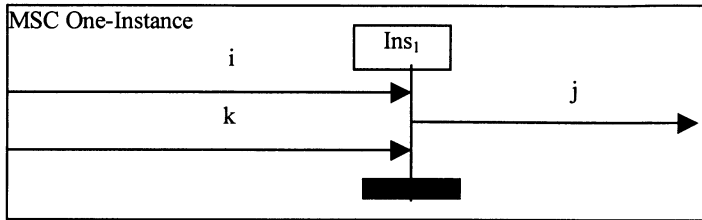


Figure 1. MSC with One Instance

The corresponding symbolism of Fig. 1 is

$$\begin{aligned} Ins &= (\Sigma_{Ins}, \Theta_{Ins}, \Psi_{Ins}, Con_{Ins}, Exp_{Ins}, Ref_{Ins}, \underline{I}_{Ins}), \\ \Sigma_{Ins} &= \{in(i, env), in(k, env)\}, & \Theta_{Ins} &= \{out(j, env)\}, & \Psi_{Ins} &= \emptyset, \\ Con_{Ins} &= \emptyset, & Exp_{Ins} &= \emptyset, & Ref_{Ins} &= \emptyset, \\ \underline{I}_{Ins} &= \{in(i, env) \bullet out(j, env) \bullet in(k, env)\} \end{aligned}$$

$$MSC \text{ One-Instance} = (I, F),$$

$$I = \{Ins\}, \quad F(\underline{I}_{Ins}) = \lambda(\underline{I}_{Ins}) = \{in(i, env) \bullet out(j, env) \bullet in(k, env)\}$$

3.2.2 MSC with Two Instances and Conditions

MSC Conditions (Fig. 2) can be defined in the form

$$\begin{aligned} Ins_1 &= (\Sigma_{Ins1}, \Theta_{Ins1}, \Psi_{Ins1}, Con_{Ins1}, Exp_{Ins1}, Ref_{Ins1}, \underline{I}_{Ins1}), \\ \Sigma_{Ins1} &= \emptyset, & \Theta_{Ins1} &= \{out(j, Ins_2)\}, & \Psi_{Ins1} &= \emptyset, \\ Con_{Ins1} &= \{c1, c2\}, & Exp_{Ins1} &= \emptyset, & Ref_{Ins1} &= \emptyset, \\ \underline{I}_{Ins1} &= \{c1 \bullet out(j, Ins_2) \bullet c2\} \end{aligned}$$

$$\begin{aligned} Ins_2 &= (\Sigma_{Ins2}, \Theta_{Ins2}, \Psi_{Ins2}, Con_{Ins2}, Exp_{Ins2}, Ref_{Ins2}, \underline{I}_{Ins2}), \\ \Sigma_{Ins2} &= \{in(j, Ins_1)\}, & \Theta_{Ins2} &= \emptyset, & \Psi_{Ins2} &= \emptyset, \\ Con_{Ins2} &= \{c1, c2\}, & Exp_{Ins2} &= \emptyset, & Ref_{Ins2} &= \emptyset, \\ \underline{I}_{Ins2} &= \{c1 \bullet in(j, Ins_1) \bullet c2\} \end{aligned}$$

$$MSC \text{ Condition} = (I, F),$$

$$I = \{Ins_1, Ins_2\}$$

$$F(\underline{I}_{Ins1}, \underline{I}_{Ins2}) = \lambda(\underline{I}_{Ins1} \parallel \underline{I}_{Ins2}) = \{out(j, Ins_2) \bullet in(j, Ins_1)\}.$$

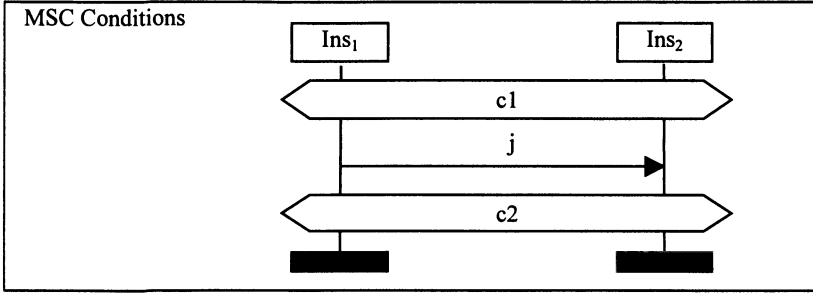


Figure 2. MSC Conditions

3.2.3 MSC with Two Instances and Inline Operator Expression

MSC Inline Expressions (Fig. 3) can be denoted in the form

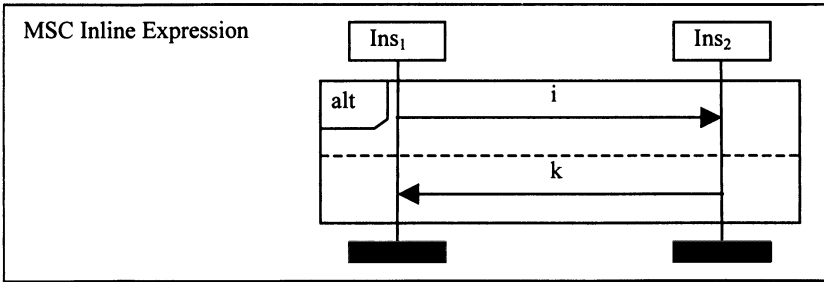


Figure 3. MSC Inline Expressions

$$\begin{aligned}
 Ins_1 &= (\Sigma_{Ins_1}, \Theta_{Ins_1}, \Psi_{Ins_1}, Con_{Ins_1}, Exp_{Ins_1}, Ref_{Ins_1}, \underline{I}_{Ins_1}), \\
 \Sigma_{Ins_1} &= \{in(k, Ins_2)\}, \Theta_{Ins_1} = \{out(i, Ins_2)\}, \Psi_{Ins_1} = \emptyset, \\
 Con_{Ins_1} &= \emptyset, Exp_{Ins_1} = \{exp(alt, [i], [k])\}, Ref_{Ins_1} = \emptyset, \\
 \underline{I}_{Ins_1} &= \{exp(alt, [i], [k])\}. \\
 Ins_2 &= (\Sigma_{Ins_2}, \Theta_{Ins_2}, \Psi_{Ins_2}, Con_{Ins_2}, Exp_{Ins_2}, Ref_{Ins_2}, \underline{I}_{Ins_2}), \\
 \Sigma_{Ins_2} &= \{in(i, Ins_1)\}, \Theta_{Ins_2} = \{out(k, Ins_1)\}, \Psi_{Ins_2} = \emptyset, \\
 Con_{Ins_2} &= \emptyset, Exp_{Ins_2} = \{exp(alt, [i], [k])\}, Ref_{Ins_2} = \emptyset, \\
 \underline{I}_{Ins_2} &= \{exp(alt, [i], [k])\}.
 \end{aligned}$$

MSC Inline Expression = (I, F),

$$I = \{Ins_1, Ins_2\}$$

$$F(\underline{I}_{Ins_1}, \underline{I}_{Ins_2})$$

$$= \lambda(\underline{I}_{Ins_1} \parallel \underline{I}_{Ins_2})$$

$$= \lambda(\{out(i, Ins_2) + in(k, Ins_2)\} \parallel \{in(i, Ins_1) + out(k, Ins_1)\})$$

$$= \{out(i, Ins_2) \bullet in(i, Ins_1) + out(k, Ins_1) \bullet in(k, Ins_2)\}$$

3.2.4 MSC with One Action and One Reference

MSC Actions and References (Fig. 4) can be denoted in the form

$$\begin{aligned}
 Ins &= (\Sigma_{Ins}, \Theta_{Ins}, \Psi_{Ins}, Con_{Ins}, Exp_{Ins}, Ref_{Ins}, \underline{I}_{Ins}), \\
 \Sigma_{Ins} &= \{in(i, Env)\}, \quad \Theta_{Ins} = \emptyset, & \Psi_{Ins} &= \{a\}, \\
 Con_{Ins} &= \emptyset, \quad Exp_{Ins} = \emptyset, & Ref_{Ins} &= \{Ref(r)\}, \\
 \underline{I}_{Ins} &= \{in(i, Env) \bullet Act(a) \bullet Ref(r)\} \\
 \text{MSC Action and Reference} &= (I, F), \\
 I &= \{Ins\} \\
 F(\underline{I}_{Ins}) &= \lambda(\underline{I}_{Ins}) \\
 &= \{in(i, Env) \bullet Act(a) \bullet Ref(r)\}.
 \end{aligned}$$

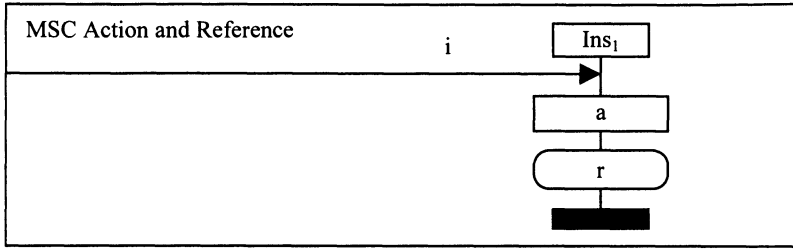


Figure 4. MSC Action and Reference

4. RELATIONSHIPS BETWEEN EFSM AND REGULAR EXPRESSIONS

This section sets up and analyzes new properties of an EFSM model, which include DEFSM, NEFSM, NEFSM with ϵ -move and the relationship between EFSM and Regular Expressions. Based on the definition of MSC, a relationship between it and Regular Expressions is introduced. Finally, the relationship of EFSM and MSC is determined. In order to achieve the above purposes, certain concepts and lemmas, used in these theorems, are introduced. All the lemmas and theorems are not proved due to the limited space available.

4.1 Characteristics and Properties of EFSM

The EFSM model differs from the traditional Finite State Machine (FSM) model in special characteristics and properties.

4.1.1 The Four Containment Relationships of EFSM

There exist equivalent relationships among the three kinds of NEFSM, while there does not exist an equivalence of DEFSM and NEFSM. According to the definitions of DEFSM and NEFSM, DEFSM can be viewed to be a special case of NEFSM-OT or of NEFSM-Tran. Moreover, NEFSM-OT and NEFSM-Tran are also special cases of NEFSM-Full, whence:

1. $\{\text{DEFSM}\} \subset \{\text{NEFSM-OT}\} \subset \{\text{NEFSM-Full}\}$,
 2. $\{\text{DEFSM}\} \subset \{\text{NEFSM-Tran}\} \subset \{\text{NEFSM-Full}\}$,
- where $\{\text{DEFSM}\}$ denotes the entire set of DEFSM.

4.1.2 Equivalence of Three Kinds of NEFSM

The equivalence of NEFSM-Tran, NEFSM-OT and NEFSM-Full will be considered next. Since $\{\text{NEFSM-Tran}\} \subset \{\text{NEFSM-Full}\}$ and $\{\text{NEFSM-OT}\} \subset \{\text{NEFSM-Full}\}$, only the following assumptions require proof:

- for any NEFSM-Tran m , there exists a NEFSM-OT m' such that $L(m)=L(m')$,
- for any NEFSM-OT m , there exists a NEFSM-Tran m' such that $L(m)=L(m')$,
- for any NEFSM-Full m , there exists a NEFSM-OT m' such that $L(m)=L(m')$.

Theorem 1: If L is a language accepted by NEFSM-Tran m , then there exists NEFSM-OT m' such that $L(m)=L(m')$.

Theorem 2: If L is a set produced by an NEFSM-OT, then there exists an NEFSM-Tran which can produce the same L .

Theorem 3: If L is a set produced by an NEFSM-Full, then there exists an NEFSM-Tran which can produce the same L .

4.2 Relationship between DEFSM and NEFSM

Unfortunately, there is not an equivalent relationship between DEFSM and NEFSM, since

Theorem 4: If L is a language, accepted by an NEFSM-OT, then no DEFSM can accept the same L .

4.3 Relationship between EFSM and Regular Expression

Based on automata theory, a Regular Expression can also define a language. By tradition, a Regular Expression *Rexp* consists only of *input* symbols. However, the EFSM model contains not only *input*, but also *output* and *task*. Thus, in the present treatment, a Regular Expression *Rexp* consists of *input*, *output* and *task* symbols; the *Behavior Sequence Set*, defined by *Rexp*, is called a language, which is denoted by $L(Rexp)$. This language is also called a *regular set*.

According to automata theory, there exist equivalent relationships between FSM, FSM with ϵ -move and *regular set* [3]. The EFSM model implies equivalence of *regular sets* and the languages accepted by NEFSM, but there is not an equivalent relationship between DEFSM and Regular Expressions. The class of $L(m)$, produced by a DEFSM m , is a sub-set of the class of $L(Rexp)$, produced by the Regular Expression *Rexp*.

Due to the equivalence between any two of the three kinds of NEFSM, all kinds of them can equivalently be denoted by Regular Expressions and *vice versa*. This conclusion is proved in

Theorem 5: Equivalence of regular expressions and NEFSM.

5. RELATIONSHIP BETWEEN MSC BEHAVIOR SEQUENCE AND REXP

A MSC Behavior Sequence denotes the order of message exchange among objects of a system. All Behavior Sequences, produced by an MSC, are called a language, accepted by MSC and denoted by $L(MSC)$.

By the definition of MSC in Section 3.2, there exist in MSC the eight basic operators: '+', '•', '||', *exp(loop)*, *exp(alt)*, *exp(par)*, *exp(opt)*, *exp(exc)*. Because *exp(alt)* and *exp(par)* have the same meanings as '+' and '||', and *exp(opt)* and *exp(exc)* is being ignored here, the number of basic operators reduces to '+', '•', '||' and *exp(loop)*.

If one imposes a limitation on *exp(loop)* and lets it loop to infinity, then the set of MSC Behavior Sequences becomes regular, a result which proves

Theorem 6: *Regular sets* and Behavior Sequence sets of MSC with limitation are equivalent.

6. RELATIONSHIP BETWEEN SDL PROCESSES AND MSC

Although the DEFSM does not cover all elements of SDL Processes, the main features of SDL Processes can be described by a DEFSM. Meanwhile each DEFSM can be denoted by a Regular Expression, whence all SDL Processes can be represented by Regular Expressions. Due to the equivalence of the regular and language sets of MSC, all Behavior Sequences produced by SDL Processes can be translated into MSC.

However, since the class of languages accepted by DEFSM is merely a sub-set of a regular set, there exists an equivalence of regular and language sets of MSC; thus the behavior sequences of MSC cannot always be translated into DEFSM. But each Behavior Sequence of MSC can be converted into EFSM as a result of the equivalence of EFSM and regular sets (EFSM includes DEFSM and NEFSM).

Hence, with regard to the translation from SDL Processes into MSC, it is complete and does not change the meaning of Message Exchange. When it comes to a conversion from MSC to SDL Processes, first translate MSC into EFSM, then check whether an EFSM is a DEFSM or not. If it is a DEFSM, then it is also a SDL Process. If it is not a DEFSM, a new structure must be defined, in order to translate NEFSM into SDL Processes.

7. CONCLUSIONS AND RECOMMENDATIONS

According to the equivalence of EFSM and Regular Expression as well as MSC Behavior Sequence and Regular Expression, SDL Processes have the equivalent meaning of an MSC Behavior Sequence for the basic components of these two languages. Hence SDL Processes and MSC Behavior Sequence can equivalently be translated from one into the other.

Based on this relationship, a further task is to define a set of rules to translate SDL Processes into MSC and *vice versa* [8, 9]. These translations will provide a more powerful and intuitive tool for system specification and description as well as overcome disadvantages of these two languages.

REFERENCES

- [1] R. Braek and O. Haugen. (1993). *Engineering Real Time Systems--An object-oriented methodology using SDL*. Prentice Hall, Englewood Cliffs.
- [2] R. Braek. (1996). SDL Basics. *Computer Networks and ISDN Systems*, Vol.28, pp.1585-1602.

- [3] J. E. Hopcroft and J. D. Ullman. (1990). *Introduction to Automata Theory, Languages and Computation*. Narosa Publishing House Reprint.
- [4] ITU Z.100. (1989). Recommendation Z.100, Specification and Description Language SDL. Geneva: ITU.
- [5] ITU Z.100. (1993). Recommendation Z.100, Specification and Description Language SDL. Geneva: ITU.
- [6] ITU-TS. (1993). Recommendation Z.120, Message Sequence Chart (MSC). *ITU-TS*, Geneva.
- [7] ITU-TS. ITU-TS Recommendation Z.120, Message Sequence Chart (MSC). *ITU-TS*, Geneva. (Publication scheduled 1997).
- [8] J. Lian and V. Wuwongse. (1999). Conversion from SDL Processes into MSC. *Technical Report*, Computer Science and Information Management Program, Asian Institute of Technology.
- [9] J. Lian and V. Wuwongse. (1999). Conversion from MSC into SDL Processes. *Technical Report*, Computer Science and Information Management Program, Asian Institute of Technology.
- [10] S. Mauw. (1996). The formalization of Message Sequence Charts. *Computer Networks and ISDN Systems*, Vol. 28, pp.1643-1657.
- [11] E. Rudolph, P. Graubmann and J. Grabowski. (1996). Tutorial on Message Sequence Charts. *Computer Networks and ISDN Systems*, Vol. 28, pp.1629-1641.

A TRADING MODEL OF STREAM BINDING SELECTION

F. Eliassen & H.O. Rafaelsen

*F. Eliassen, University of Oslo, Dept of Informatics, P.O. Box 1080, 0316 Oslo, Norway,
E-mail frank@ifi.uio.no*

*H.O. Rafaelsen, University of Tromsø, Dept of Computer Science, 0937 Tromsø, Norway,
E-mail hansr@cs.uit.no*

Key words: Multimedia, QoS, Typing, ODP

Abstract: Distributed multimedia information systems require a range of different interaction styles ranging from simple remote operation interaction to complex patterns of interaction involving both discrete and continuous data. The standardized reference model for Open Distributed Processing (ODP) defines a binding model that encapsulates different interaction styles within explicit binding objects. In this paper we introduce the idea of using a trading model for selecting potential binding types. In this scheme a set of binding factories are located based on a specification of the required properties of the binding. The located set of binding factories are all capable of instantiating binding objects with properties conforming to those specified by the client.

1. INTRODUCTION

The notion of stream interface has been proposed as the preferred means to convey media streams in distributed multimedia systems (Coulson et al, 1992). A stream interface consists of a collection of source and/or sink media flows. Stream interfaces have been adopted in the work on Open Distributed Processing (ITU-T Draft Recommendation X.901, 1995) and TINA-DPE (TINA-C, 1995) and more recently by OMG (Object Management Group, 1996).

The act of stream binding establishes a logical association between compatible stream interfaces for the purpose of exchanging continuous flows as dictated by the type and direction of the flows. During binding, the interfaces to be bound must be type-checked for compatibility. Informally, type-checking means ensuring that the properties of each source flow are as expected by the corresponding sink flow.

In RM-ODP stream bindings are explicit. This supports direct client control of the binding during its lifetime. In figure 1, a binding object connects four interfaces by means of local bindings that associate the interfaces of the objects with the interfaces of the binding object.

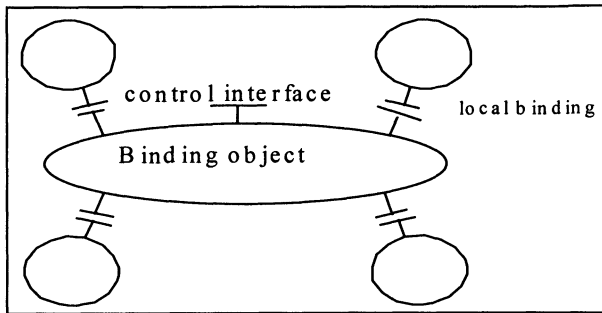


Figure 1. The explicit binding model

A binding type (or template) defines a particular class of binding objects by identifying the type of interfaces which can participate in the binding, the roles they play, and the way behaviour at the various binding interfaces are linked (Lindsey & Linington, 1995). Binding factories are objects that create bindings. In our model, a binding factory is associated to a binding type such that by invoking the factory object's `create` method, a new binding of the associated type is created.

Through the local bindings the binding object receives and delivers information for binding participants according to the causality and type of the bound interfaces. Type checking as explained above, is applied when adding a new interface to the binding. If an object with interface α is offered to fulfil a role β , then the type of α must be compatible with the type of β .

In this paper we present foundational concepts for applications to select binding types and associated binding factories according to their requirements based on a trading model. In this scheme a set of binding factories are located based on a specification of the required properties of the binding. The located set of binding factories have in common that they are all capable of instantiating binding objects with properties conforming to those specified by the client. Such a facility will, for example, allow

evolution of supported binding types in a distributed computing environment without sacrificing support of existing applications. This can be ensured by requiring that a new version of a binding type must *conform* to the older version. Other areas that might benefit from a binding type selection facility as indicated above, are mobile applications that need to trade binding types satisfying their requirements in changing environments, and multimedia databases where the required properties of the binding used to present a query result are generally not known in advance, but rather depends on the specifics of the query (Plagemann et al, 1999).

The remainder of the paper is organized as follows. In section 2, we offer further motivation for our work. Following that, in section 3 we then present the main ideas of our binding framework MBS (Model of Bindings and Streams). Section 4 introduces the notion of binding type and offers a definition of a conformance relationship of binding types. An application of this relationship is illustrated through examples. Section 5 concludes and discusses some further work.

2. TYPE CHECKING AND BINDING OF STREAM INTERFACES

The act of binding includes type-checking of the interfaces to be bound for compatibility. Compatibility rules for stream interfaces, however, have been deemed outside the scope of the RM-ODP standard (ITU-T Draft Recommendation X.901, 1995); the proposed rationale for which is that the optimum form of subtyping will depend on the application. Although this is a valid observation, the advantages of detecting type errors at compile (or bind) time, rather than at run-time, are no less important for stream interfaces than they are for operational interfaces. In the work of TINA-C, the need for a compatibility relationship for stream flows that is more relaxed than equivalence, is recognized, but no definition is offered (TINA-C, 1995). A proposal for audio/video support in CORBA (IONA et al, 1997), also introduces the notion of flow end-point compatibility. This determines whether two flow endpoints are connectable. In their proposal two flow endpoints are compatible if they share a common protocol over which they can transport media, and if they share a common format for media encoding/decoding (e.g. MPEG). Hence QoS parameters are not considered.

Microsoft's ActiveMovie framework (Microsoft ActiveMovie, 1996) also includes the notion of "compatibility negotiation" between "pins" (connection points that carry flows between different processing objects). The subject of this negotiation is data compatibility rather than QoS.

In this work, however, we adopt and extend a generic type model for stream flows and associated type checking rules earlier proposed in (Eliassen & Nicol, 1996) into a generic model of bindings and streams (MBS). In contrast to many of the above approaches, the model of (Eliassen & Nicol, 1996) is open-ended and can in principle support any set of flow parameters. An implementation of the flow type model and examples of its application are described in (Eliassen & Mehus, 1998)

In the flow type model, a flow type is specified by indicating the media type of the flow such as audio or video, its causality (source or sink), and a set of quality attributes such as rate and resolution. Furthermore, an attribute value is specified as a set of "atomic" values. This will in general enhance the chances of successful binding. For example when a sink flow type specifies a set of different names on the video encoding attribute, it actually declares that it can accept flows where the video can have any of the indicated formats. A flow type specification is interpreted as a set of potential stream flow qualities such as data formats and temporal constraints. This supports the definition of a variety of flow type relationships based on set theory. For example, compatibility is determined by computing the set of common flow qualities supported by the two endpoints. Compatibility requires that this set is not empty. If the two endpoints can support more than one common flow quality, a flow quality negotiation protocol may be employed to choose the actual flow quality to be used.

A further feature of the type model of (Eliassen & Nicol, 1996) is that a flow specification may include "individual" requirements to the satisfaction of compatibility such as optional configurations of media elements that may occur in the flow and media element dependencies. In this paper we extend this feature to stream interfaces allowing optional configurations of flows and flow dependencies to be specified. For example, a stream interface modeling an access point to a video conference provider, may exploit this feature to express the alternative audio and video flow configurations and qualities that can be supported. Requiring compatibility ensures that a joining conference participant satisfies at least one of the alternatives supported by the provider.

QML is a recent proposal for a QoS specification language (Frølund & Koistinen, 1998). The semantics of QML is similar to our stream and flow type model but its applicability is demonstrated for operational interfaces only. Other work that considers QoS specifications and/or negotiations includes (Campbell, 1996), (Dini & Hafid, 1997) and (Zinky et al, 1997). However, these works do not provide anything corresponding to a type model of streams and bindings, including type relationships such as subtype, compatibility and conformance.

3. MODEL OF BINDINGS AND STREAMS (MBS)

For the stream abstractions, we have developed a *generic* model called MBS (Model of Bindings and Stream Interfaces). In MBS, a binding object may support a number of stream interfaces. Each interface has an associated role in the binding. As in (Lindsey & Linington, 1995) a role is specified by its name and stream interface type. For example, a multicast video binding would typically support a producer role and a consumer role. In some cases the binding will support operations for adding interfaces to a binding in a named role. A stream interface consists of a collection of source and/or sink media flows.

3.1 Flow Type Model

A flow type is specified by indicating the media type of the flow such as audio or video, its causality (source or sink), and a set of quality attributes such as rate and resolution. Furthermore, an attribute value is specified as a *set of* "atomic" values. For example when a sink flow type specifies a set of different names on the video encoding attribute, it actually declares that it can accept flows where the video can have any of the indicated formats. The following example of an H.261 video flow type features optional playback rates.

```
type VideoPhone = Flow [
  V:Video[encoding:H.261, rate:2..24]];
  A:Audio[encoding:PCM, rate:{8000,16000}]]
(V & A) | A
```

This specification states that a VideoPhone flow consists of two different element types labeled V and A respectively. Each element type includes a declaration of the generic media type such as Video or Audio, and a specific set of attributes, referred to as a media descriptor, specifying quality properties of the element type. The expression (V & A) | A is referred to as the structural constraints of the flow (Eliassen & Mehus, 1998). It specifies legal configurations of flow elements that may occur in an instance of the flow. The above structural constraint indicates that an instance of a flow can consist of either video and audio elements, or audio elements only. Hence we may think of the specification as modelling an adaptable flow endpoint.

A flow type specification is interpreted as a set of potential flow qualities (QI) and flow configuration (SI) that can be produced by a source flow endpoint or is acceptable to a sink flow endpoint. This interpretation of a flow type allows us to define a variety of flow type relationships based on

set theory. The quality interpretation of a flow type is defined as the combination of the interpretation I of each of its element types (for further details see e.g. (Eliassen & Nicol, 1996)).

A flow type M is a *subtype* of the flow type N if both the quality and structural interpretation of M is a subset of the corresponding interpretations of N . We derive that $M = \text{Flow}[A;B]$ is a *strict quality subtype* of $N = \text{Flow}[C;D]$, denoted $M <_q N$, if $I(A) \subseteq I(C)$ and $I(B) \subseteq I(D)$. On the other hand, a *relaxed quality subtype* may support fewer element types than the super type such that, for example, $M = \text{Flow}[A]$ is a relaxed quality subtype of $N = \text{Flow}[B;C]$, denoted $M <\sim_q N$, when $I(A) \subseteq I(B)$. Finally, M is a *structural subtype* of N denoted $M <_s N$, if $SI(M) \subseteq SI(N)$.

Our approach for type checking binding attempts is to require that the source and the sink can at least support one common flow quality. This relationship we refer to as *quality compatibility*. Informally, two flow types are *strict quality compatible* (denoted $<>_q$) if their respective element types have pair wise non-empty set intersection of their respective interpretations. We may for example conclude that $\text{Flow}[A;B] <>_q \text{Flow}[C;D]$ if $I(A) \cap I(C) \neq \emptyset$ and $I(B) \cap I(D) \neq \emptyset$. Two flow types M and N are *relaxed flow quality compatible* (denoted $<\sim>_q$) if a subset of the element descriptors in each of the two flow types, are pair-wise compatible. Thus if $I(A) \cap I(C) \neq \emptyset$, $\text{Flow}[A;B] <\sim>_q \text{Flow}[C;D]$ even if B and D are incompatible, i.e. $I(B) \cap I(D) = \emptyset$. Two flows of type are *structural compatible*, denoted $M <>_s N$, if their structural interpretations have non-empty intersection. Different variants of the compatibility relationship where some variants are weaker than others, are the following:

- i) fully strict compatible, if $M <>_q N$ and $M <_s N$.
- ii) partially strict compatible, if $M <>_q N$ and $M <>_s N$.
- iii) fully relaxed compatible, if $M <\sim>_q N$ and $M <_s N$.
- iv) partially relaxed compatible, if $M <\sim>_q N$ and $M <>_s N$.

3.2 Stream Type Model

In MBS the notion of configuration constraint allows for the specification of individual requirements to the satisfaction of compatibility for stream interfaces in terms of alternative combinations of flows that may be configured in a stream binding. A stream configuration constraint is written as a structural constraint over flow labels. The set of alternative configurations of flows of a stream interface is referred to as its structural intersection SI .

Two stream interfaces S and T are *compatible*, denoted $S <> T$, if S and T have a common legal configuration such that the flows in these

configurations of **S** and **T** are pairwise compatible. Consider, for example, the following stream interface type of a video conference binding type

```
type videoConfProducer = stream [
  a : sink flow [
    a1: Audio[encoding:PCMA,
              rate:{8000,16000}]]];
  v : sink flow [
    v1: Video[encoding:H.261,
              rate:(2..24)]]]
configurations a|(a&v) ] //end stream
```

and the interface type `audioTalk` offered by a potential participant of the binding

```
type audioTalk = stream [
  a : source flow [
    a1: Audio[encoding:{PCMA,GSM},
              rate: 8000]]; //end flow
configurations a ] //end stream
```

The above interface types are compatible.

The MBS *conformance rules* express conditions for substitutability of stream interfaces. If the stream interface **T** *conforms* to the stream interface **S**, then **S** may be replaced transparently by **T**. A stream interface **T** conforms to a stream interface **S** if and only if for every flow of **S** there is a corresponding flow in **T** such that the flow of **S** is a subtype of the flow of **T** and $SI(S) \subseteq SI(T)$.

The following example illustrates a case where a video source `mpgSource` is upgraded to support additional playback rates, video encodings and audio, all encapsulated in the stream interface `mpg_mjpgSource` such that `mpg_mjpgSource` conforms to `mpgSource`.

```
type mpgSource = stream [
  mpgFlow : source flow[
    h: Video[encoding:mpeg,rate:{20,25}]]
configurations mpgFlow ] //end stream

type mpg_mjpgSource = stream [
  mpgFlow : source flow [
    h: Video[encoding:mpeg,rate:{20,25,30}]]];
  mjpgFlow : source flow [
    hj: Video[encoding:mjpeg,rate:{20,25,30};
    au: Audio[encoding:{PCMA,GSM},rate:8000]]
configurations mpgFlow|mjpgFlow ] //end stream
```

4. SELECTION OF BINDING TYPE

In this section we present an approach for selecting binding types from binding type requirement specifications. Our approach for specifying binding types is similar to the RIVUS template language (Lindsay & Linington, 1995), the main difference being that binding requirements are specified using the stream interface type model introduced in this paper.

4.1 Specifying binding types

A binding type is defined as a 5-tuple $\langle T, P, M, \Delta, E \rangle$ where T denotes a set of role types, P a set of roles, M a set role matching requirements (one for each role), Δ a set of role causalities, and E a set of role cardinality requirements.

A *role type* τ is defined as a set of stream interface types, i.e. $\tau = \{T_1, \dots, T_n\}$. A *role* defines binding object roles and is specified as a role name and a role type, $r: \tau$. For example, a video conference binding type could define the roles `talk` and `listen` where the role `talk` could be of the role type `{videoConfProducer}` (c.f. section 3.2).

Role matching requirements specify for each role of the binding type the kind of type matching required when an interface is offered to fulfil the role. The kind of matching that can be specified is either a subtype or a compatibility relationship. Thus we model role matching requirements as a set of pairs $\{\langle r_1, m_1 \rangle, \dots, \langle r_n, m_n \rangle\}$ where r_i is a role name and m_i is a match kind.

Binding behaviour defines *causalities* between instances of roles. We model role causality as a tuple $\langle C, r_1, r_2, m \rangle$ where C specifies a causality option, r_1 and r_2 are roles, and m specifies whether conversion between r_1 and r_2 is supported by the binding. Conversion is supported if $m = \text{conv}$, otherwise $m = \text{no\_conv}$. Conversion is required if the roles r_1 and r_2 have incompatible roles types, and may be required in those cases where alternative behaviour is specified at the corresponding interfaces (e.g. alternative encodings) and the chosen behaviour for r_1 is allowed to be incompatible with the chosen behaviour for r_2 . For example, if the type of r_1 and r_2 is both `video`, and the type `video` offers both `mpeg` encoding and `mjpeg` encoding, and r_1 chooses a configuration with `mpeg`, and r_2 chooses a configuration with `mjpeg`, then a conversion between `mpeg` and `mjpeg` is needed. This might, for example, be realised as a suitable transcoder running within an active network.

As in (Lindsay & Linington, 1995) we define three options for how roles can be mapped together. Specifying $\langle \text{ONE-ONE}, r_1, r_2, m \rangle$ means that the

binding object creates a one to one mapping between a single instance of role r_1 and a single instance of role r_2 , while $\langle \text{ONE-MANY}, r_1, r_2, m \rangle$ means that the binding object creates a one to many mapping between a single instance of role r_1 and all instances of role r_2 . Finally, $\langle \text{MANY-MANY}, r_1, r_2, m \rangle$ means that the binding object creates a mapping between all instances of role r_1 and r_2 .

Role cardinality is a specification of the number of a particular role the binding object can support (Lindsey & Linington, 1995). It is modelled as a pair $\langle r, i \rangle$ where r is a role and i is a range where the minimum value states the number of the role that is needed for the binding object to make sense, while the maximum states the largest number of the role the binding object is willing to support.

$\langle \text{talk}, 2..10 \rangle$

4.2 Binding type conformance

An application selects a binding type by stating binding type requirements to a trader that compares the requirements to binding type offers. A binding type requirement specification is with one exception only, identical to a definition of a binding type as outlined above, while a binding type offer is simply a binding type specification. Selection is based on a conformance relationship between binding type requirements and binding type offers. The result of the selection is the identification of a set of binding factories that are all capable of instantiating binding objects with properties conforming to those specified by the client. A conformance relationship for binding types must be based on conformance of stream interfaces and notions of role matching, role causality, and role cardinality satisfaction.

While a role matching requirement of a binding type offer is specified as a pair $\langle r, m \rangle$, the role matching requirement of a binding type requirement is specified as a triple $\langle r, m, \sigma \rangle$ where σ indicates whether a stricter role matching requirement than m is acceptable ($\sigma = \text{narrow}$) or not ($\sigma = \text{no\_narrow}$). It is easy to show that strict subtype logically implies ($\Rightarrow$) all other match kinds, relaxed subtype logically implies fully and partially relaxed compatibility, full compatibility logically implies partial compatibility, and strict compatibility logically implies relaxed compatibility. Thus, $\langle r, \text{relaxed\_compatibility}, \text{narrow} \rangle$ is satisfied by $\langle r, \text{full\_compatibility} \rangle$.

Definition 1 (*role matching satisfaction*) A role matching requirement $\langle r_1, m_1 \rangle$ of a binding type offer, satisfies a role matching requirement $\langle r_2, m_2, \sigma \rangle$ of a binding type requirement if and only if $m_1 = m_2$, or $\sigma = \text{narrow}$ and $m_1 \Rightarrow m_2$. $\square$

Definition 2 (*role causality satisfaction*) A role causality $\langle C, r_1, r_2, m \rangle$ is satisfied by a role causality $\langle C', s_1, s_2, n \rangle$ if and only if $C = C'$, s_1 conforms to r_1 , s_2 conforms to r_2 , and if $m \neq n$, then $n = \text{conv}$. $\square$

Definition 3 (*role type conformance*) A role type $\tau = \{T_1, \dots, T_n\}$ conforms to a role type $\sigma = \{S_1, \dots, S_n\}$ if and only if there exists a bijection β between τ and σ such that for all $(T_i, S_j) \in \beta$, T_i conforms to S_j . $\square$

Definition 4 (*binding type conformance*) A binding type offer $B_1 = \langle T_1, P_1, M_1, \Delta_1, E_1 \rangle$ conforms to a binding type requirement $B_2 = \langle T_2, P_2, M_2, \Delta_2, E_2 \rangle$ if and only if there exists a bijection β between the sets of role causalities Δ_1 and Δ_2 such that for all $(\delta, \varepsilon) \in \beta$ with $\delta = \langle C_1, r_1, s_1, m_1 \rangle$ and $\varepsilon = \langle C_2, r_2, s_2, m_2 \rangle$, δ satisfies ε , and the role cardinality requirements $\langle r_1, i_1 \rangle$ and $\langle s_1, j_1 \rangle$ in E_1 satisfies the corresponding role cardinality requirements $\langle r_2, i_2 \rangle$ and $\langle s_2, j_2 \rangle$ in E_2 such that $i_2 \subseteq i_1$ and $j_2 \subseteq j_1$, and the role matching requirements $\langle r_1, m_1 \rangle$ and $\langle s_1, n_1 \rangle$ in M_1 satisfies the corresponding role matching requirements $\langle r_2, m_2, \sigma_2 \rangle$ and $\langle s_2, n_2, \mu_2 \rangle$ in M_2 . $\square$

Example 1. The following is a simple specification of a binding type requirement, for an audio conference binding. We will later refer to this binding type by the name `audioConf`. We define two role types `audioConfProducer` = {audioTalk} and `audioConfConsumer` = {audioListen} where

```
type audioTalk = stream [
  a : sink flow [
    a1: Audio[encoding:{PCMA},rate: 8000]]
  configurations a ] //end stream

type audioListen = stream [
  a : source flow [
    a1: Audio[encoding:{PCMA},rate: 8000]]
  configurations a ] //end stream
```

The corresponding binding roles are

```
talk : audioConfProducer
listen : audioConfConsumer
```

Note that the causalities of the flows are specified as they are provided by the binding, i.e. a source flow of an audio conference participant (a talker) must locally bind to a corresponding sink flow offered by the binding.

The role matching requirements are

```
<talk,partially_relaxed,narrow>
<listen,partially_relaxed,narrow>,
```

while the role causality required by the audio conference application is

```
<MANY-MANY,talk,listen,no_conv>,
```

and the role cardinality requirements are <talk,2..6> and <listen,2..6>.

A binding type offer that satisfies the above binding type requirement is specified below. We will later refer to this offer by the name AVConf. The offer defines two role types AVConfProducer={AVTalk} and AVConfConsumer={AVListen} where

```
type AVTalk = stream [
  a : sink flow [
    a1: Audio[encoding:{PCMA,GSM},
      rate:{8000,16000}]]];
  v : sink flow [
    v1:Video[encoding:H.261,
      rate:(2..24)]]]
  configurations a|(a&v) ] //end stream

type AVListen = stream [
  a : source flow [
    a1: Audio[encoding:{PCMA,GSM},
      rate:{8000,16000}]] ];
  v : source flow [
    v1: Video[encoding:H.261,
      rate:(2..24)]]]
  configurations a|(a&v) ] //end stream
```

The binding roles of the offer are

```
gen : AVConfProducer
rcv : AVConfConsumer
```

The role matching requirements are

```
<gen,fully_strict_compatible>
<rcv,fully_strict_compatible>,
```

while the role causality requirement of the binding type offer is

```
<MANY-MANY,gen,rcv,conv>,
```

and the role cardinality requirements are <gen,2..20> and <rcv,0..20>.

It can easily be shown that the binding type offer satisfies the binding type requirement according to definition 4. We basically need to show that

the role causality requirements of the binding type offer, $\langle \text{MANY-MANY, gen, rcv, conv} \rangle$, satisfies to role causality requirements of the binding type requirement, $\langle \text{MANY-MANY, talk, listen, conv} \rangle$, and that role cardinality and role matching requirements of corresponding roles are satisfied. $\square$

Below we give a further example of how binding types and stream interface types can be used as a foundation for creating explicit bindings.

Example 2. Suppose two users, User_A and User_B, want to engage in a teleconference. Their capabilities are given by their stream interface specifications.

User_A supports two stream interface types: videoConfProducer, specified in section 3.2; and videoConfConsumer specified below.

```
type videoConfConsumer = stream [
  a : sink flow [
    al: Audio[encoding:PCMA, rate:{8000,16000}]]];
  v : sink flow [
    vl:Video[encoding:H.261, rate:2..24]]
  configurations a|(a&v) ] //end stream
```

User_B supports the stream interface types: audioTalk, specified in section 3.2; and audioListen specified below.

```
type audioListen = stream [
  a : sink flow [
    al: Audio[encoding:{PCMA, GSM}, rate:{8000}]]
  configurations a ] //end stream
```

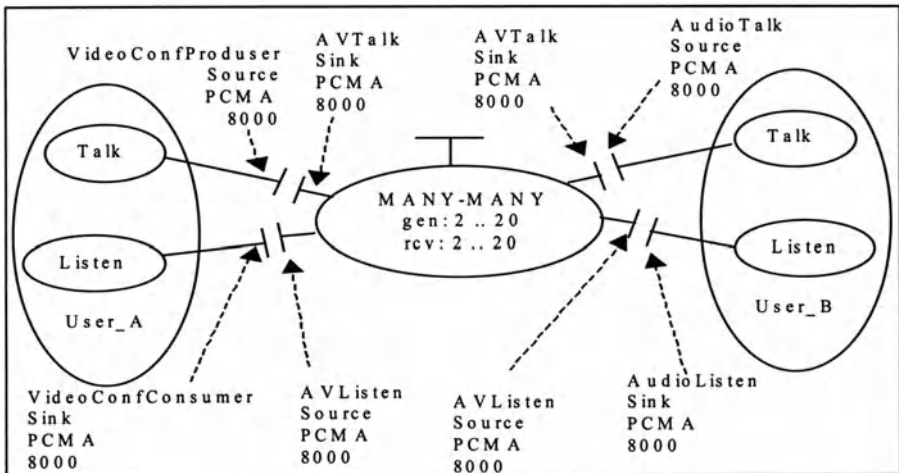


Figure 2. A conference binding

Suppose User_B initiates the teleconference by requesting a reference to an appropriate binding factory from the binding trader. In this case we assume that User_B generates the binding type requirement, `audioConf`, as specified in example 1, by comparing the interface types supported by both users.

For the sake of the example we further assume that the distributed processing environment in this case does not offer a binding exactly as specified by User_B. However, the environment offers a binding with the properties of `AVconf`, as specified in example 1.

Since the offered binding `AVconf`, conforms to the requested `audioConf`, the offered binding can be used. Figure 2 shows how the specifications of User_A and User_B are used to create a explicit binding. It is possible for both users to attach to the binding, since the binding provides compatible stream interfaces. New participants might join the conference by invoking operations at the bindings control interface. The binding will check that the new participants interfaces are compatible with the offered interfaces of the binding. □

5. CONCLUSIONS AND FUTURE WORK

In this paper we introduced foundational concepts supporting mechanisms for selecting appropriate explicit stream bindings based on statements of binding type requirements provided by the application. We introduced the idea of using a trading model for selecting potential binding types, and showed how an earlier proposed type model for stream flow can be extended to support stream interfaces and binding type selection based on a notion of binding type conformance.

In our current work we focus on extending the type model for supporting the situation where binding objects specify alternative behaviour at its supported interfaces and it is needed to negotiate the actual interface behaviour to be used. Our approach is to introduce the notion of *policy specifications* that can be associated to each flow of a stream interface. A policy specification effectively specifies an order on the interpretation of a flow type. This ordering can, for example, be used to select QoS parameters for the flow based on a given resource availability, and for adapting the quality of the flow by taking the flow interpretation order as the specification of the preferred degradation path. The results of this work will be reported elsewhere.

REFERENCES

- Campbell, T. (1996). A Quality of Service Architecture. *PhD Thesis*, Lancaster University
- Coulson, G., Blair G. S., Stefani, J. B., , Horn, F., Hazard, L. (1992). Supporting the Real-Time Requirements of Continuous Media in Open Distributed Processing. *Technical Report MPG-92-35*, Lancaster University.
- Dini, P., Hafid, A. (1997). Towards Automatic Trading of QoS Parameters in Multimedia Distributed Applications, *In proceedings of IEEE/IFIP ICODP/ICDP Conference*, Toronto, Canada, 166 - 179
- Frølund,, S., Koistinen, J. (1998). Quality-of-Service Specification in Distributed Object Systems, *Distributed Systems Engineering Journal*, Vol.5, No.4
- Eliassen, F., Nicol, J. R. (1996). Supporting Interoperation of Continuous Media Objects. *Theory and Practice of Object System*, Vol.2, No.2, Wiley, 95-117.
- Eliassen, F., Mehus, S. (1998). Type Checking Stream Flow Endpoints. *Middleware '98*, The Lake District, England, 16-18 Sept, Chapman & Hall, 305 - 322.
- Lindsey, D., Linington, P.F. (1995). RIVUS: A Stream Template Language for Capturing Multimedia Requirements, *Lecture Notes in Computer Science (LNCS 1052)*, Springer Verlag, 259 - 277
- Microsoft. (1996). Microsoft ActiveMovie: Software Development Kit, Beta Release, June 1996.
- ITU-T X.901 | ISO/IEC 10746-1. (1995). ODP Reference Model Part 1: Overview. *Draft International Standard*.
- IONA Technologies, Pic, Lucent Technologies, Inc, Siemens-Nixdorf, AG. (1997). Control and Management of A/V Streams Request for Proposal. OMG RFP Submission, *OMG Document: telecom/97-05-07*.
- Object Management Group. (1996). Control and Management of A/V Streams Request for Proposal. *OMG Document: telecom/96-08-01*.
- Plagemann, T., Eliassen, F., Göbel, V., Kristensen, T., Rafaelsen, H., Adaptive QoS Aware Binding of Persistent Multimedia Objects, In proceedings of International Symposium on Distributed Objects and Applications (DOA'99), Edinburgh 5-6 sept 1999 (to appear)
- TINA-C. (1995). TINA Object-Definition Language, Version 1.3. *TINA-C Deliverable*.
- Zinky, A., Bakken, D.E., Schantz, R.D. (1997). Architectural Support for Quality-of-Service for CORBA Objects, *Theory and Practice of Object Systems*, Vol.3, No.1, Wiley

USING ROLES WITH TYPES AND OBJECTS FOR SERVICE DEVELOPMENT

Rolv Bræk

NTNU department of Telematics, and

Sintef Telecom and informatics,

Trondheim Norway

e-mail: rolv.braek@informatics.sintef.no

Key words: Roles, Object-orientation, Service models, Interface models

Abstract: This paper is about achieving open-ended flexibility using roles, types and objects. Unfortunately the term “role” is used to mean different things in different approaches, and it is not always clear how it relates to other important notions such as types or classes and objects. The first concern of this paper is therefore to clarify what roles are. The second is to outline how they can be utilised to achieve open-ended flexibility in systems and services. It does not go deeply into the technical details, but concentrates on overall principles and how they may be utilised in combination with existing methods and languages, in particular UML [OMG 99], [Rumbaugh 99], SDL [ITU 93] and MSC [ITU 96b].

1. INTRODUCTION

Due to increasing user expectations, ICT convergence, de-regulation and the vast possibilities of the ubiquitous Internet, the ability to quickly develop and deploy new services will become an increasingly important competitive factor in the future. Whether the solution will be a “smart” or a “stupid” network may be debated to lengths, but there is no question that flexibility to provide an open range of services will be a key feature. Considering the wide range of multimedia services enabled by new technology, safe and rapid service development will pose a formidable challenge. Even in traditional “single services networks” where “services” are seen as features

of a basic service, e.g. the basic telephone call, this has been difficult to achieve. “Intelligent Networks” were introduced to improve the service flexibility in such networks, but are inherently constrained by the underlying basic call and network assumptions, and not well suited for the emerging “open services network”. It is not reasonable to believe that service logic in the future will fit into a monolithic pre-defined framework, like the IN architecture. It is more likely that service logic will be provided by open-ended “object communities” that are allowed to evolve and change more freely as long as the result is useful and without undesirable effects. This paper is about achieving such open-ended flexibility using roles, types and objects.

2. ABOUT ROLES, TYPES AND OBJECTS IN GENERAL

According to normal terminology, objects are phenomena; i.e. tangible entities with substance and behaviour, while types⁴ are concepts representing the features shared by all objects belonging to the type and roles are properties that may be associated with both objects and types.

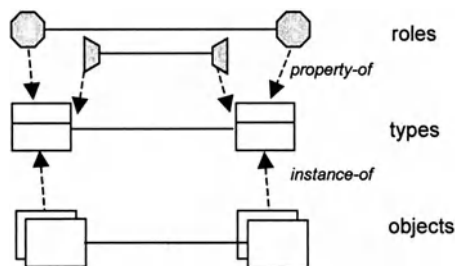


Figure 1. Roles, types and objects

2.1 Roles

The notion of role is used extensively in everyday talk either to characterise objects in relation to other objects, like “father”, or to indicate their function/responsibility in an organisation, like “project leader”.

A classical example of the functional use is found in the theatre. The ensemble of a theatre consists of actors (i.e. objects) while the plays they perform are defined in terms of roles the actors shall enact during the play.

⁴ We use the term “Type” in the same meaning as “Class”.

Roles are described (in plays) independently of particular actors, but in order to be performed, they must be assigned to actors. Actors, on the other hand exist independently of particular roles, and are able to play many different roles. Note that roles do not fully specify the actor behaviour, but give room for interpretation allowing actors to put some of their own personality into the play.

Family relationship is a classical example of relational roles. “Father”, “son” and “cousin” all denote roles that a person may play in relation to other persons. Each person is normally able to play many different relational roles, and will do so slightly differently from other persons, depending on their personality.

Clearly these two notions of role are very useful when seeking to describe and understand reality, and therefore they have gradually made their way into modelling techniques and methods. UML for instance, uses roles in both meanings. An AssociationEnd has a role name, which denotes a relational role. A ClassifierRole represents the role of an object in a collaboration, which is a functional role, see [OMG 99], [Rumbaugh 99].

Relational roles will be termed *association roles* in the following, in keeping with UML terminology⁵. They originate from data modelling, in particular Entity-Relationship (E-R) modelling, where they have been used to name roles for the endpoints of relationships, with only informal semantics carried by the name.

Functional roles will be termed *service roles* in the following to indicate that they represent the part an object plays in a service, function or task. Informal use of service roles is found in many strands of life, for instance to denote roles in a human organisation such as “team-leader”, “secretary”, or “accountant”. Within telephone engineering, it has been common to describe telephone services using role names like the “A-subscriber” and the “B-subscriber”. Service roles are central to the OORAM approach [Reenskaug 95], and partly covered by ClassifierRoles in UML Collaboration diagrams. In UML, several ClassifierRoles may be associated with a Classifier, but a ClassifierRole can bind to only one Class, so ClassifierRoles are not really independent concepts that can be re-used in different Classes. If ClassifierRoles were to model service roles in general, this UML constraint would have to be removed.

Obviously a Class must provide all the operations and accept all signals specified in its Classifier roles, but apart from this neither UML nor OORAM deals with role behaviours and how role behaviours compose into Class behaviours in terms of state machines.

⁵ Strictly speaking UML reserves the term “AssociationRole” for associations linking ClassifierRoles in Collaboration diagrams.

In the SISU method [Bræk 93] the notion of *role-behaviour* was introduced to give dynamic semantics to interface roles. A role-behaviour was used to define the observable behaviour of an object at an interface. The original idea was that behaviour roles would help to simplify validation of interfaces between objects by reducing the state space needing to be explored. In order to be faithful to the full object behaviour, roles have to be *input consistent*, and it turned out that lack of input consistency was a strong indicator that errors like unspecified reception and deadlock would occur in any context where objects of the type were used. A simple rule was formulated that could be checked for each type with very limited effort, even manually, without needing to compose objects and performing state space exploration. Moreover it was possible to use the rule constructively to avoid introducing errors in the first place.

The overall use of roles in a methodology was further elaborated in TIME [Bræk 99], but TIME did not elaborate on role-behaviours.

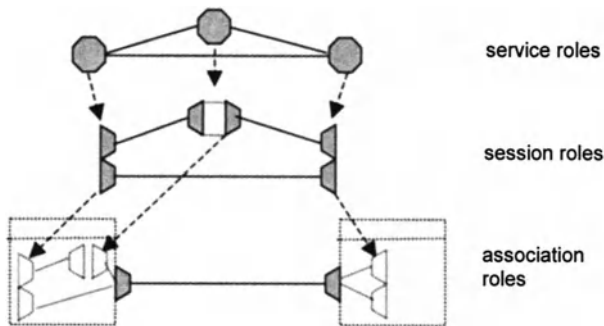


Figure 2. Illustration of role relationships

Here we shall use 3 notions of *role-behaviours*, see Figure 2:

- An *association role* describes the visible behaviour of one object at an association end. A special case is an *interface role*, which describes the visible behaviour at an interface. Note that it is possible for several associations to use the same interface, and in that case, the interface role will be a composition of all association roles on that interface.
- A *service role* describes the visible behaviour of an (anonymous) object participating in a service, function or task.
- A *session role* (or *service association role*) is the visible behaviour of one service role at one association end. A special case is an *interface session role*. Note that a given service role may have several associations and interfaces with other service roles. Each of these has a session role.

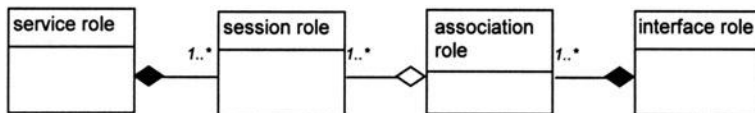


Figure 3. Role relationships

All these roles can be seen as subsets of the object properties observable from different angles like projections. Just like projections, roles are useful in several ways:

- *Architecture definitions.* Interfaces are central to system architectures, and often more important than types, because interface roles can be used to define interfaces precisely without binding the types of objects that uses the interface. This is a central point in a Plug-and-Play architecture. Static interface definitions like those of CORBA IDL, UML and SDL is a necessary minimum, but not sufficient to guarantee that objects will inter-work correctly in all cases. A better guarantee can be given if interface definitions also specify the dynamic interface role-behaviour.
- *Reuse.* Roles are reuse entities in their own right. This follows from the independence between roles and objects and the fact that a given role sometimes is to be provided by many different objects and types. All types accessing a given interface, for instance, must provide the same interface roles. Roles can also be used as search criteria to find types and objects that can play given roles, for instance as part of a trader's functionality.
- *Design synthesis:* When designing a type, the roles it shall provide serve as specifications for the type design. It should be noted here that role-behaviours only specify behaviour *properties*. They are not behaviour design units. Design synthesis requires that information is added, and that roles are composed. Therefore the insight of a human designer is needed in most cases, although precisely defined role-behaviours may greatly facilitate the design process. Behaviour design synthesis is an open research area where partial solutions exist, but general approaches are yet to be found.
- *Design verification.* A given design should be verified against the roles it shall provide. This means either to check that the roles can be derived from the design by making projections, or to check that the roles will be performed by the design. As explained in [Bræk 93], roles also enable useful checks on the internal design consistency to be performed.

- *Validation of associations and links*⁶. Association roles can be used like plugs and sockets to validate associations (between types) and links (between objects). It has been shown in [Bræk 93] that interface roles help to reduce the state space of a reach-ability analysis.

2.2 Types (or classes) and roles

A type (or class) represents the features that all objects considered as its instances have in common. A type definition will normally have two main parts: a context part defining the external associations and interfaces, and a content part defining the internal construction in terms of attributes, operations and behaviour. This general pattern for a type definition is illustrated in *Figure 4*.

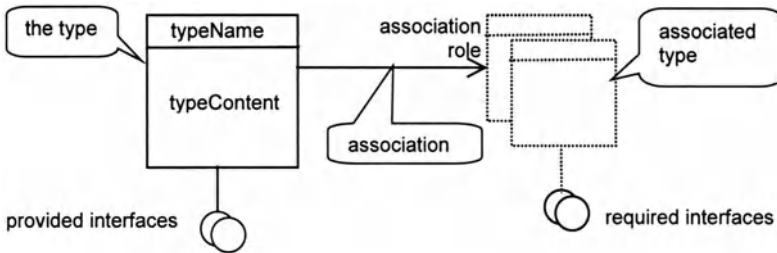


Figure 4. Aspects of a type illustrated using UML notation

This picture is implicit in the UML approach where a type is modeled as a Class. Its static context and content are defined using Class diagrams, while the dynamic behaviour is defined using State Machines. Associations relate a Class to other Classes, and AssociationEnds have roles. An AssociationEnd in UML represents a set of objects of the target Class (the class it connects to) and implicitly describes properties of the source Class (the Class at the other end of the Association). In addition to this, an AssociationEnd (role) defines requirements on the set of target objects that can validly be linked to a source object. AssociationEnds can identify required interfaces, but there is no notation to describe the behaviour in connection with interfaces. An Interface in UML represents a collection of input operations⁷.

⁶ In accordance with UML terminology, associations are relationships between types (classes), while links are relations between objects.

⁷ one would expect that signals were part of the interface, but this is not allowed according to UML v1.3, beta.

Note that in UML, the target Class of an AssociationEnd is specified, and this may be overly restrictive. More flexibility can be obtained by specifying only the *required roles* without binding the Class, since this will allow instances of any Class to be linked as long as the required roles are satisfied. In an open evolving system where new types may be included as the system evolves, this is more appropriate.

The purpose of a type is often related to a set of roles (both service roles and association roles) that its instances shall play. Different types may play a given set of roles differently. The content part of the type will define exactly how the roles will be *provided*. In addition to providing roles, a type will *require* that (instances of) associated types play given roles.

This leads up to a relationship between behaviour roles and types as illustrated in Figure 5.

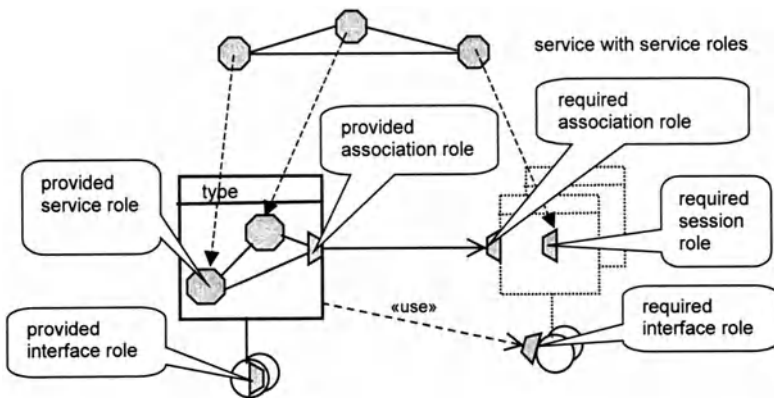


Figure 5. Roles related to a type illustrated using UML

Note that this view is relative to the type being defined. Provided roles are the roles that objects of the type can play. Required roles are the roles they expect other objects to play.

The set of provided roles (that an object can play) is called its *role repertoire*.

In UML Class diagrams, Associations and Interfaces serve mainly to give properties to the Classes (types) defined by the diagram. Still it makes sense to check the consistency of Associations and Interfaces. For an Association between two Classes (types) to be valid, the provided roles must *contain* the required roles on both ends. Similarly for interfaces: the provided interface roles must contain the required interface roles. Statically this means to check that the required sets of signals and operations are contained in the provided sets. Dynamically it means to check that the required role behaviours are

contained in the provided role behaviours. (What this means will not be elaborated here, but it is akin to checking that a Message Sequence Diagram, MSC [ITU 96b], can be executed by a design, see e.g. [Kristoffersen 91], [Nahm 91].)

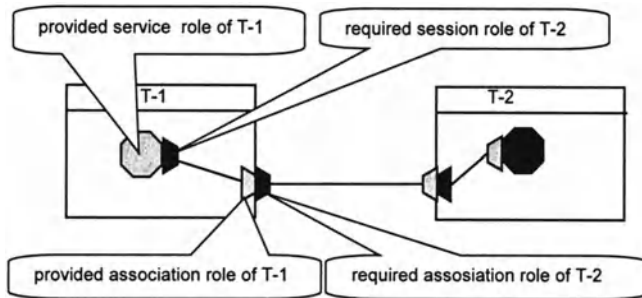


Figure 6. Aligning provided and required roles: required roles shall be contained by provided roles

Although the purpose of a type is to play roles, its content is not composed from roles. The principal reason is that roles, by nature, are incomplete and partial views on objects, and are not construction units. They are properties that shall be satisfied by the construction, and not parts of the construction itself, just as a set of 2D drawings define views on a 3D construction without being part of it.

If roles are to be used actively in behaviour design, like 2D drawings are used in geometrical design, it is essential that the correspondence between roles and designs become well defined. Roles should map to sets of possible design units so that type designs can be defined by selecting and composing such units into a whole that is both internally consistent and compliant with the roles. (Just as in geometrical design.)

This is not common practice today, partly because the concept of behaviour roles is quite new, and partly because the correspondence between behaviour roles and behaviour design units is not well understood. To the extent that roles are used in type design at all, it is normally an ad-hoc approach where the correspondence between roles and design units is lost and not utilised further after the initial design is made. This is a pity because roles seem to have a potential for life-long benefits that could be utilised better if the correspondence between roles and designs were better understood.

2.3 Objects and object structures

Object structures consist of objects and links between objects, where objects are instances of types and links are instances of associations.

When a link is created, it is necessary that the required (association and service) roles are contained in the provided roles. Otherwise the link will be invalid and exceptions may result. Therefore, when a link is created it is necessary for roles to be aligned so that containment is ensured. We consider three levels of *role alignment*:

- *Validation*. Checking that the required roles are contained in the provided roles, and raising exceptions if not.
- *Adaptation*. Negotiating and adapting the roles within bounds given by the object's type. This is performed by some protocols today.
- *Learning*. Learning to play new roles by receiving role implementations when existing roles are insufficient. Downloading Java Applets can be seen as one way of doing this.

Ideally, role alignment should be performed dynamically whenever a link is created, before it is being used. This is not common practice, however. It is more common to rely on off-line design analysis before the system is created combined with error detection and exception handling during use.

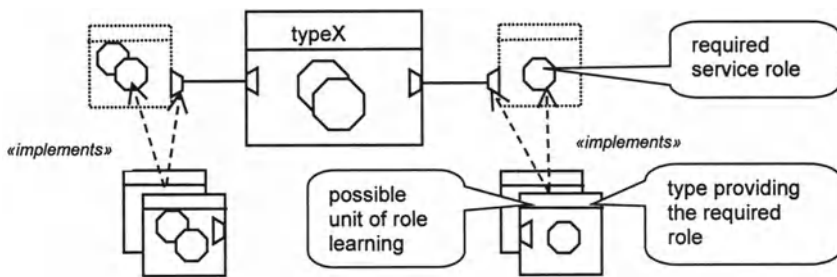


Figure 7. Using types to accomplish role learning

In the dynamic world of future open services ICT systems, this will be too inflexible and also too error prone, especially if new services and types are to be introduced dynamically. We expect that role learning will become a very important technique in such systems. Role learning can be accomplished even if the general problem of role design units and role composition is not fully solved. It is sufficient that a set of (alternative) types exist that can provide the roles, and that instances of one of these may be installed on demand in the learning object, see Figure 7. In this way downloading/installing objects of predefined types accomplish role learning. The role implementation may be adapted to the receiving actor, e.g. to a

terminal with limited capabilities, by selecting the appropriate object type to download/install.

In general we only want object structures that serve their purpose and are well formed in some way. Rules for well formed structures can be classified as *content rules* and *context rules*. Content rules govern the internal composition of (instances of) a type. A (context-free) grammar, for instance, is entirely made up of content rules. Context rules, on the other hand, govern the external use of (instances of) a type. Gate constraints in SDL and interface definitions in CORBA IDL, can be seen as context rules. Traditionally content rules have been more used than context rules. One reason is that content rules are simple to make, another reason is that context rules have not been strong enough to ensure well-formed structures alone. Even CORBA IDL definitions are insufficient because they do not cover dynamic interface behaviour. Our idea of required and provided roles is intended to remedy this.

Content rules restrict the possible structures more than is necessary or desirable in many cases. OMG has realised this, and therefore the cornerstone of CORBA and component-based development in general, is context rules in the form of interface definitions. By only using context rules, it is possible to achieve a higher degree of flexibility, and open-endedness. If the only constraints are context rules given for the component types, any object structure is well formed as long as all context rules are satisfied. This is similar to the “Lego principle” where the possibilities are limited only by the “rules of the bricks”. Ideally, context rules should be checked for each composition step and if satisfied they should guarantee that the resulting system is well formed. Unfortunately the *static* interfaces of CORBA-IDL, UML and SDL do not guarantee correctness. Using provided and required roles combined with role alignment is intended to improve the situation. Given a set of types, any structure of objects will be well formed as long as every object plays the behaviour-roles that other objects in the structure require from them, and this will be a stronger criterion for well formed-ness than possible with static interface definitions like IDL.

3. SERVICE DEVELOPMENT USING ROLES

Figure 8 shows the main models that we assume are developed for an (abstract) application system (not considering physical implementation and performance). The first step is to define the services using service models with service roles. The next step is to map service roles to types and to design the types. Finally the object system is created from the types.

Similar overall approaches have been used in the industry for years, e.g.: by Ericsson and Alcatel, but without utilising roles, types and context rules in the way we propose here.

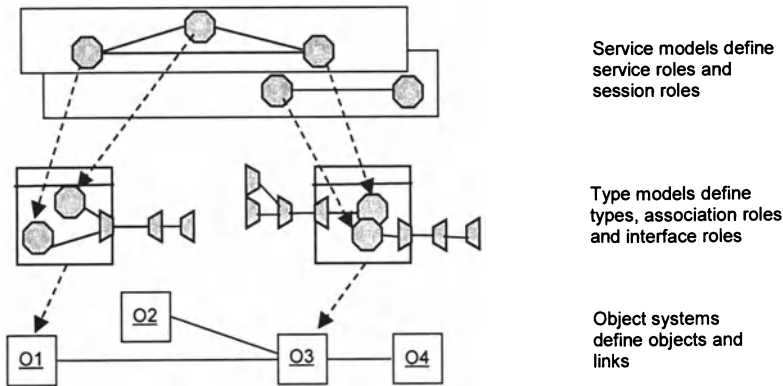


Figure 8. Models and objects in service development

Making each separate service model is usually neither very difficult, nor very resource demanding. Each service model may consist of:

- UML collaboration diagrams that identify the structure of service roles, but without binding the roles to Classes.
- A set of UML Sequence diagrams or MSC diagrams that (partially) define the overall service behaviour in the form of interactions.
- Possibly some SDL process graphs or UML Statechart diagrams defining the complete behaviour of service roles and session roles. In these diagrams interactions that are not part of a role are hidden, showing only interactions that are part of the role. In order to represent transitions that are triggered by invisible inputs, such inputs are represented by designated input symbols, i.e. *none* in SDL.

The special thing here is to use SDL process graphs or UML Statechart diagrams to define role behaviours precisely.

Difficulties start in type design when service models are mapped on types. This is the challenging part that we seek to improve. In order to improve we must understand the fundamental nature of the relationship between services, roles and types:

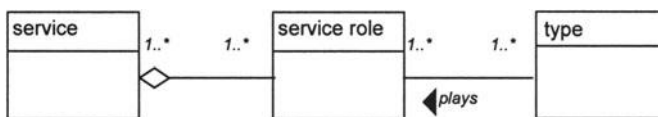


Figure 9. Service-type relationship

- Typical services involve more than one service role, and the roles will be mapped on different types. Therefore a service change typically will impact several types.
- Some types must provide several service roles.
- Many different types may provide a given role.
- Service roles are not independent but interact, both within a service and between different services. The so-called service interaction problem (or feature interaction problem) is related to undesirable interactions between different services (or instances of the same service).
- Service roles may be assigned to objects dynamically when dynamic links are created and may contend for the same objects (actors). Creating dynamic links often involves some kind of resource allocation, and therefore dynamic role assignment is akin to resource allocation. In this light, a resource allocator may be seen as an actor allocator or actor manager/trader. It will respond to requests for actors that can play some given role, perhaps negotiate the role (e.g. CallForward in stead of Busy), and ensure that undesirable interactions are avoided. In [Fritsche 95], a scheme to detect feature interactions by means of service roles was proposed.

Due to the n:m association between services and types, a service change is likely to impact several types, and this is a main reason why service flexibility has been so hard to achieve.

The solution to these problems must be found primarily in type designs and the principles used to implement roles in types. One possible approach is to define *design patterns* corresponding to roles, and define how these patterns can be elaborated and composed into correct designs. The resulting design may be seen as *personality* implementing the roles.

The personality determines the way that roles are combined and realised. It may add some flavour to the interpretation of roles as long as the resulting behaviour stays within the bounds of the roles. Exactly what this means will not be explained here, but the relationship between roles and personality is similar to the relationship between specifications and implementations.

In order to simplify and speed up service development, practicable principles for role design patterns and pattern composition will be of great help, not only to design types, but more importantly to allow roles to be learned dynamically on demand.

In traditional designs, objects have a fixed type with a fixed role repertoire. If the type of an object is allowed to change dynamically, the object will be more adaptable to changing requirements. Changing the type means, in most cases, changing the roles, so type changes is one way to adapt objects to changing roles. Another way would be to change the roles without changing the type. This could provide even more flexible

adaptations, but it would require a type concept that supported changing roles. In effect it would require *actor* types that were able to “learn” new roles dynamically, as proposed in [Aagesen 99]. This, in turn, would require that roles had “implementations” that could be composed dynamically into a whole, something like manuscripts that can be interpreted along with other manuscripts. (If such manuscripts can be devised, a possible implementation can be Java Applets/Beans.) Such manuscripts may be developed from role design patterns.

One step towards more flexibility, can be to partition the set of service roles into *provider-roles* and *client-roles*. The intuition is that provider roles are those that provide the services, while client roles are those that use them. In traditional telecommunication system designs both kinds of roles have been treated the same way, and therefore a service change has impacted types playing provider roles and client roles alike. Typically a new end-user service has required design changes to the user interface as well as to the central service logic, and sometimes even changes to the protocols. Our idea is that client roles can be learned dynamically by client objects as part of role alignment, and thereby reduce the need to redesign the types of the client objects. Much of the success of the www can be attributed to the general ability of client browsers to adapt to server functionality (through the transferred html, Java applets, etc.).

The proposed principle is to limit the number of types that must be manually changed to those playing provider roles, using role learning for client roles. This requires a type design that:

- simplifies adding or changing provider service roles in its role repertoire;
- can hold descriptions, or references to descriptions, of required client roles in a form that enable client objects to fetch role implementations as part of role alignment;
- can itself learn client roles required from other objects as part of role alignment.

4. CONCLUSIONS

The purpose of this paper has been to clarify different notions of roles and indicate how they may be used in service development. It assumes that services in the future will be implemented by flexible object structures that can change and adapt dynamically to market needs. An open-ended approach where service modules can be developed and combined in very flexible ways, by different suppliers, to accommodate completely new multi-media services is clearly needed. One idea presented here is to associate required and provided behaviour roles with types, so that provided and required roles

may be aligned when objects of the types are linked together. Not only does this enable an incremental validation of links to take place, it also opens the possibility that objects may learn to play roles dynamically and thereby dynamically adapt to new services and situations. Dynamic role learning is seen as an important future technique to facilitate service development.

REFERENCES

- [Bræk 93] R. Bræk, Ø Haugen. (1993). *Engineering Real Time Systems - An Object Oriented Methodology using SDL*, Prentice Hall, ISBN 0-13-034448-6.
- [Bræk 99] R. Bræk et al. (1999). Quality by construction exemplified by TIME – The Integrated Method. *Teletronikk*, 95:1, ISSN 0085-7130.
- [Fritsche 95] N. Fritsche. (1995). Runtime Resolution of Feature Interactions in Architectures with separate Call and Feature control. *Feature Interactions in Telecommunications Systems III*, IOS Press, ISBN 90 5199 238 6.
- [ITU 93] ITU-T. (1994). Recommendation Z.100 ITU Specification and Description Language (SDL). *ITU-T*, ("SDL-92").
- [ITU 96b] ITU-T. (1996). Recommendation Z.120 Message Sequence Charts (MSC). *ITU-T*, ("MSC-96").
- [Kristoffersen 91] F. Kristoffersen. (1991). Message Sequence Chart and SDL Specification Consistency Check. *SDL'91 Evolving Methods, Proceedings of the Fifth SDL Forum*, North Holland: Elsevier.
- [Nahm 91] R. Nahm. (1991). Consistency Analysis of Message Sequence Charts and SDL Systems. *SDL'91 Evolving Methods, Proceedings of the Fifth SDL Forum*, North Holland: Elsevier.
- [OMG 99] Object Management group. (1999). *OMG Unified Modeling Language Specification* (Draft) version 1.3 beta R1.
- [Reenskaug 95] T. Reenskaug, P. Wold, and O. A. Lehne. (1995). *Working With Objects*. Manning: Prentice Hall.
- [Rumbaugh 99] J. Rumbaugh, I. Jacobson, G. Booch. (1999). *The Unified Modeling Language Reference Manual*. Addison-Wesley, ISBN 0-201-30998-X.
- [Aagesen 99] F. A. Aagesen et al. (1999). Towards a plug and Play Architecture for Telecommunications. *Proceedings of the Fifth IFIP Conference on Intelligence in Networks* (this issue).

A REFERENCE MODEL FOR CONCEPTUALISING THE CONVERGENCE OF TELECOMMUNICATIONS AND DATACOMMUNICATIONS SERVICE PLATFORMS

Pasi Kemppainen

Helsinki University of Technology, Otakaari 1, FIN-02150 Espoo, FINLAND, Tel. +358 9 451 2174, Telefax +358 9 451 5014, E-mail Pasi.Kemppainen@intellitel.com

Key words: Convergence, Network Intelligence, Reference model, Service platforms, IP integration, Telecommunications infrastructure

Abstract: There is no doubt that the telecommunications industry is experiencing a major metamorphosis. Telecommunications, data communications and content from different media and services are converging, resulting in web-based shopping services, mobile interactive banking services and personalised content services, to name a few. This paradigm shift, the convergence, is the driving force of the next generation technologies, services and even businesses. To system vendors and service operators these much-desired new end-user services bring unprecedented business opportunities. However, to bring the best of all these worlds together, there are major technological challenges to meet within the communications network and service infrastructure. And to rationalise and explain the convergence of the existing and forthcoming discrete network infrastructures and services, new conceptualisation approaches are needed. This paper introduces the concept of Telecommunications Network Infrastructure Reference Model (TNI RM) for providing a holistic view for the convergence of telecommunications and datacommunications networks and systems. In addition, the TNI RM presents the importance of content networks in the ongoing convergence.

1. INFRASTRUCTURE CONVERGENCE: THE IP INTEGRATION

1.1 The Next Generation Telecommunications Infrastructure

When compared to data communications networks, telecommunications networks are complicated and diversified in nature. Practically, it can be stated that modern data communications networks are relying on IP (*Internet Protocol*) based technologies. Yet while the underlying infrastructure can be built with IP-independent technologies such as DSL (*Digital Subscriber Line*) access, SDH (*Synchronous Data Hierarchy*)⁸ in transport and ATM (*Asynchronous Transfer Mode*) and Ethernet in switching, IP is the middleware which integrates the whole infrastructure realising the pervasive service network called the Internet.

But telecommunications networks are constructed from a wide variety of protocols and architectures. There are presently no universal middleware protocols to integrate these networks. Telecommunications networks range from WLL (*Wireless Local Loop*) to cable in access, from IN (*Intelligent Networks*) to GSM (*Global System for Mobile Communications*) in value-added services and from TMN (*Telecommunications Management Network*) to proprietary systems in management networks. All make use of architectures and signalling of their own. Therefore the telecom technologies are highly segmented, and this is why there still is no single and universal telecommunications service network yet.

According to the *Telecommunications Network Infrastructure Reference Model* (TNI RM) (Figure 1), telecommunications networks can roughly be divided into seven categories based on their functional and infrastructural characteristics:

- Access networks,
- Transport and bearer networks,
- Switching and signalling networks,
- Service networks,
- Operations and management networks,
- Business networks, and
- Content Networks

⁸ Also known as SONET (Synchronous Optical Network) in North America.

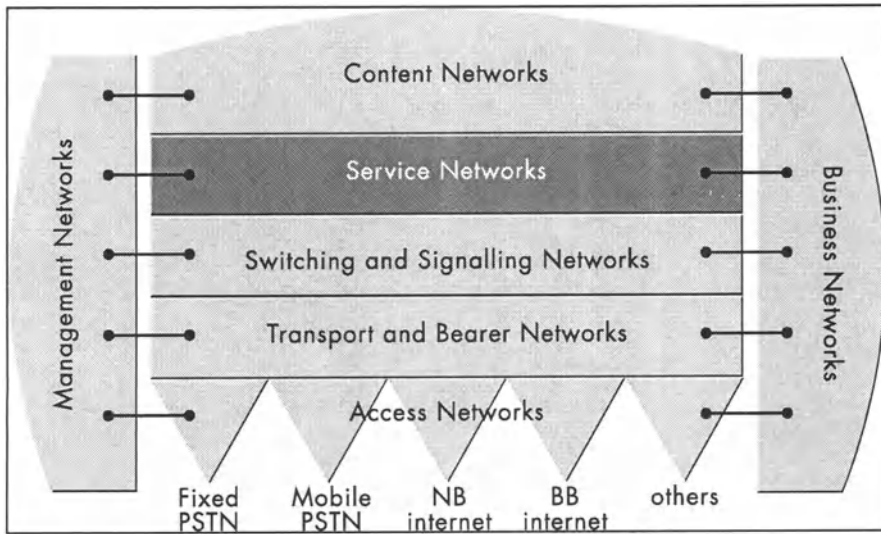


Figure 1. The Telecommunications Networking Infrastructure Reference Model (TNI RM)

1.2 The Networking Convergence: IP-integration

Obviously, current telecommunications networks inherently include some or all of aforementioned functionality tightly coupled, as in GSM networks. This is also true in traditional telephone networks (PSTN), where all networking and services (*Plain Old Telephony Services*, POTS) can be managed by one monolithic, and often proprietary, architecture. However, telecommunications infrastructures and service competition, catalysed by global deregulation and liberalisation, will unavoidably lead to the infrastructure rationalisation, the aforementioned networks will begin to emerge more distinctly.

In fact, to address the infrastructure rationalisation, telecommunications operators are already building their core networks with multiple access technologies ranging from xDSL to cable networks, and implementing mobile network services integrated with fixed network services. Finally different business and management systems are connected to these networks in abundance to realise the necessary business operations (Figure 2).

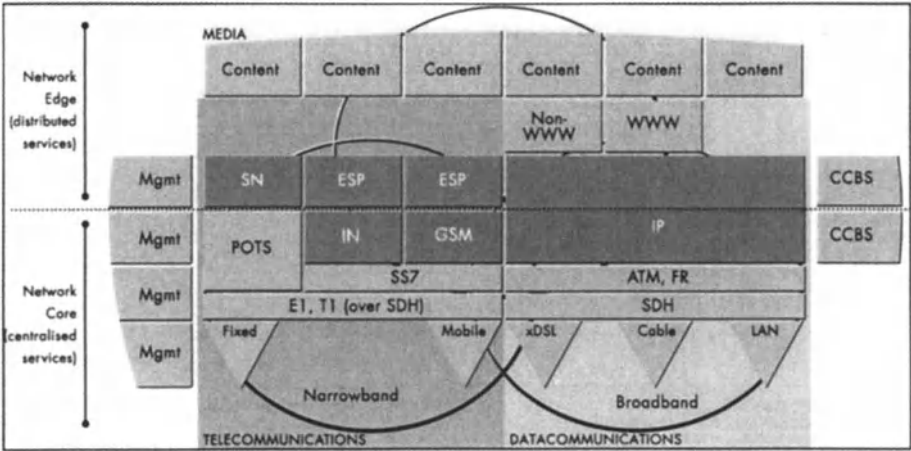


Figure 2. Phase 1 of NI evolution. The networking multiplicity: multiple networks, multiple service platforms, multiple service content formats, multiple business (CCBS⁹) and management systems

It can be claimed that the engine for realising the networking convergence is about making IP technologies the service network middleware for next generation telecommunications infrastructure. However, this is not to say the Internet will replace traditional voice-based telecommunications networks. Instead it will bring new dimensions to communicate ranging from web-based mobile interactive information services based on WAP (*Wireless Application Protocol*) to multimedia conference services based on the *Voice over IP* (VoIP) technologies. Therefore, the convergence technologically means the deployment of leading edge Internet technologies and architectures, which can be termed as the IP intelligence; first conjointly with the existing telecommunications architectures (Figure 3), and then in the core in the emerging technologies and architectures (see Section 2).

⁹ Customer Care and Billing Services

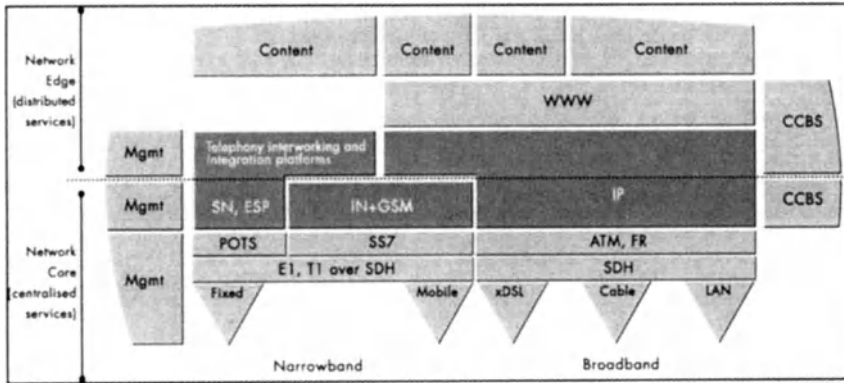


Figure 3. Phase 2. The IP-integration: WAP, VoIP, GSM-IN -integration (CAMEL¹⁰), XML¹¹ based content provisioning, content aggregation (portals and hubs)

The outcome of the IP integration will be the emergence of highly sophisticated and ubiquitous services connecting the disparate networks by utilising IP technologies. To further address and describe the consequences and challenges of this IP dominated infrastructure revolution, we will introduce the concept of the *Network Intelligence* (NI).

2. SERVICE PLATFORM CONVERGENCE: THE FRAMEWORK OF NETWORK INTELLIGENCE

2.1 Network Intelligence: What is it?

Despite its relatively recent discovery, the framework and paradigm of *Network Intelligence* (NI) infrastructure has gained different emphases. The common factor in these emphases has been that they concentrate basically on the service platforms. These platforms neither implement transport nor switching architectures, nor do they specify management or business operations networks. In this way, Network Intelligence, as the name implies, is all about making the present and future converged services more intelligent by integrating separate networking paradigms. In addition, the concept of NI explains new complex networking inter-service functionality and interfacing, such as brokering and agent-based systems, resulting in more proactive and autonomous converged networks. Therefore, it can be

¹⁰ Customised Applications for Mobile Enhanced Logic

¹¹ eXtensible Markup Language

said that NI applies a synergistic approach to the convergence: it integrates the networks at the services level, but the outcome is bigger than the sum of its parts.

The objective of the NI integration is based on the fact that as telecommunications and Internet-based services converge to become more access independent, the service platforms must adhere to this trend as well. These NI platforms concurrently support multiple protocols and network interfaces and can simultaneously control multiple converged services (Figure 4). Example of NI integrated service will be GPRS based services, which require the integration of Internet-based switching and GSM based call control and subscriber management.

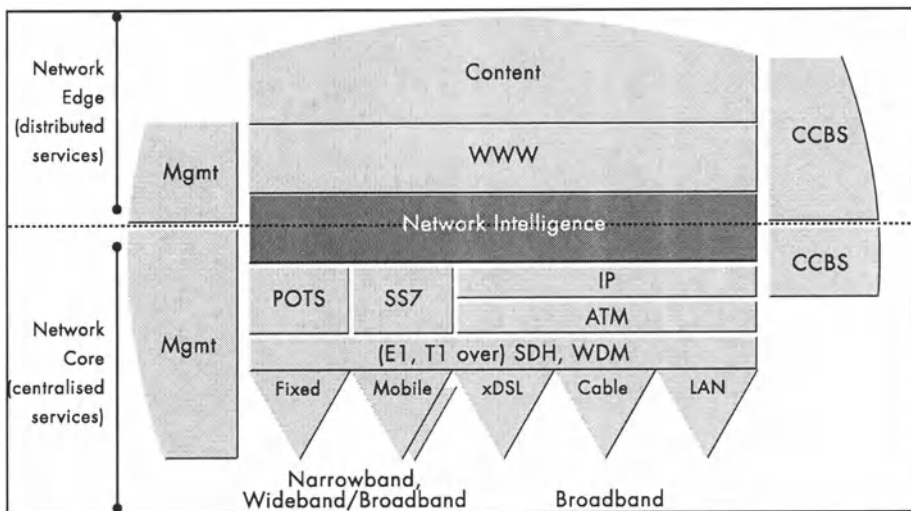


Figure 4. Phase 3. Network Intelligence integration, or service platform consolidation: WWW-based multicontent services with multiple access, converged services billing, GPRS, Network Intelligence Middleware solutions

Although the NI integration will enable seamless service provisioning over multiple access networks, this is not enough. Services still require multiple layers to interface with, such as SS7 and IP based interfaces in signalling and PSTN and ATM based interfaces in switching, and the service platform itself must control the signalling and switching integration. For example, the VoIP gateways integrate the telecommunications and IP networks by low level packet conversions, and adapt the call control and database access across the networks.

The key driver and core component for the inevitable infrastructure consolidation is expectedly the IP (Figure 5). In fact, there are technologies

available to integrate the PSTN and SS7 networks with IP based services such as the SUN JAIN (*Java Advanced Intelligent Network*) [1] interface specification for SS7 Java component encapsulation and British Telecom Parlay [2] interfaces for third party IP based PSTN services. Furthermore, in the emerging next generation telecommunications standards, UMTS (*Universal Mobile Telecommunications Service*) [3] and TINA (*Telecommunications Information Networking Architecture*) [4] in particular, service platforms are inherently based on IP. So, the ultimate end-result of the service platform consolidation will be a relatively clean convergence of telecommunications and IP technologies and networks bringing about a universal service and content control for multiple access networks, and thus appliances.

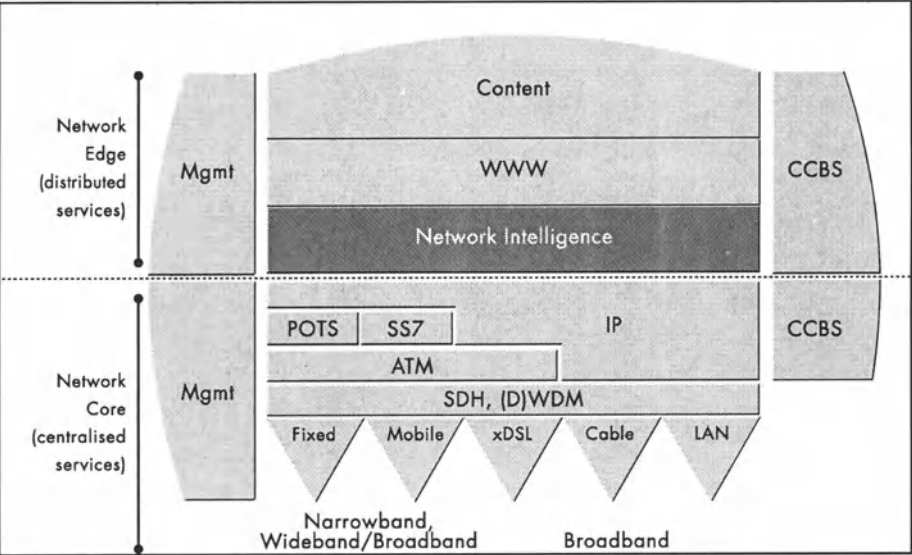


Figure 5. Phase 4. UMTS and TINA: IP-based networking interfaces, Internet-based converged service deployment, spontaneous networking such as IBM T Spaces [5] and SUN Jini [6], agent and brokering technologies

However, though there are service platforms featuring some of the required integration functionality as mentioned, their emphases vary quite widely. The next Section discusses the characteristics and emphases of the present and emerging service platforms catalysing the convergence.

2.2 Network Intelligence Platform Characteristics

For the NI integration, at least three different yet complementing service platform emphases (in addition to the IP based platforms) can be introduced:

- Telecommunications, IN and GSM in particular, enhanced **value added services platforms** (emphasis on market differentiation) (*Evolution Phases 1 and 2*)
- Telecommunications and Internet-based **telephony interworking and integration platforms** (emphasis on unified messaging and overlay switching) (*Phases 2 and 3*)
- Telecommunications and Internet **service infrastructure consolidation and convergence platforms** (emphasis on NI integration) (*Phases 3 and 4*).

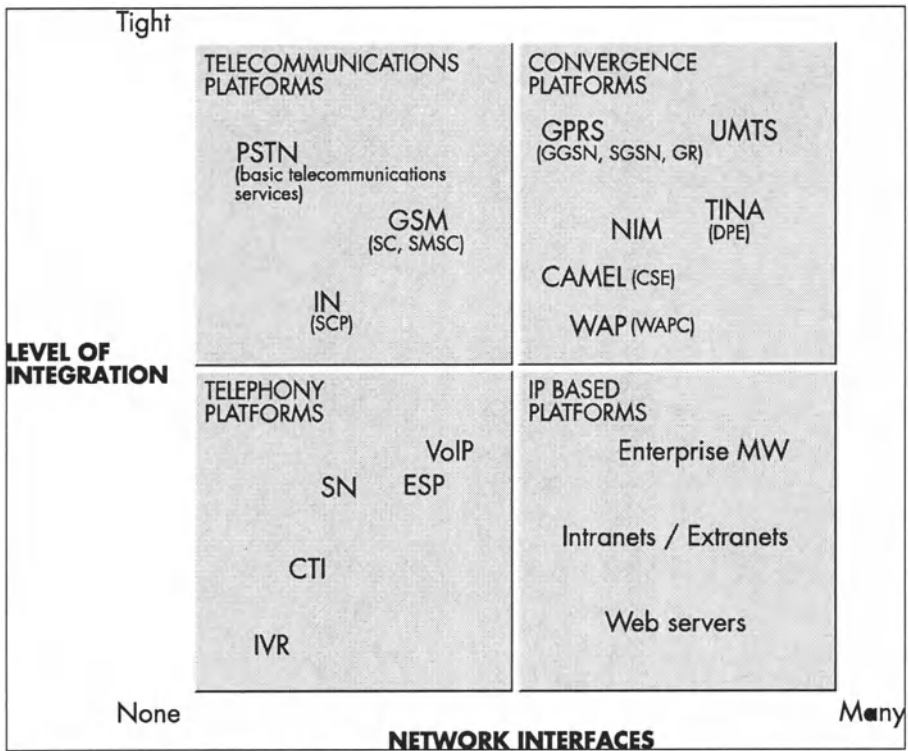


Figure 6. The NI platforms by category.

As presented in Figure 6, the paradigm of NI is not about IN network functionality integrated within GSM systems, nor is it plainly Internet-based

telephony solutions extended with telecommunications network interfaces. It is an overall framework for provisioning present and next generation telecommunications services seamlessly in fixed (including the Internet) and mobile networks.

The challenge then is not to divide and conquer all of the service networks, but merely having the ability to understand and interface with the present and future telecommunications network systems and services.

2.2.1 Value Added Services Platforms

Originally, the concept of Network Intelligence was coined by IN system vendors in the mid-90's, apparently to differentiate from the traditional IN and GSM systems and services. The primary objective has been to introduce value-added services for operators to attract the customers to subscribe to the premium-priced services and to gain customer loyalty. Good examples of new breed value-added services are enhanced corporate VPN services, like mobile centres, and premium priced GSM short message-based services, like banking and news services.

As aforementioned earlier, the foundation of the NI concept was based mainly on the existing IN and GSM systems, employing the traditional switches and service control and creation systems. Later on, the concept had been expanded to integrate the Internet services, particularly WWW, with the basic fixed and mobile network services. The objective has primarily been to enable subscribers to parameterise telecommunications services through WWW.

2.2.2 Telephony Interworking and Integration Platforms

Another more recent NI platform emphasis had been introduced largely by *Computer Telephony Integration* (CTI) and *Enhanced Service Platform* (ESP)-vendors. The CTI-IN platforms take more pragmatic perspective to integrate the IN networks with the Internet-based CTI- and PSTN-based IVR-systems (*Interactive Voice Response*), such as *Service Nodes* (SN), to form unified messaging systems. The ESP-IN integration is mainly based on the overlay switching implemented with commercial programmable switches and service nodes. The objective is to offer cost-efficient and rapid time-to-market enhanced telecommunications services especially for new fixed network service operators (*Competitive Local Exchange Carriers*, CLECs) and small and medium sized mobile operators.

The essence of the aforementioned platforms is to integrate the telephony call control and service control functionality. The objectives are basically to

enable subscribers to control the telephony functions such as the voice mail and faxes from their desktops, and to distribute the telephony applications, such as the email-to-speech conversion, over the Internet. Furthermore services, particularly the SNs or ESPs based enhanced services, are implemented largely in fixed networks for various payment and billing services. However, the most rapid growth has been in mobile services, like GSM prepaid services and fraud control management services.

Perhaps the most touted recent telephony interworking based technology has been *Voice over IP* (VoIP). The objective of VoIP technologies is to enable Internet-based telephony services, including IN-type call routing services. The ultimate goal is to integrate the Internet and telecommunications networks to form universal communications network. Unsurprisingly, various vendors and standardisation bodies have defined different VoIP architectures and services.

The IETF has several working groups focusing on the using the Internet to control PSTN services. The Voice over IP, VoIP group [7] concentrates on voice and multimedia transmission, conversion and control aspects between the networks utilising the ITU-T H.323 standards. The *PSTN/Internet Interworking* (PINT) group's [8] objectives are to define and recommend the usage of the Internet to control PSTN services. The group is focusing on defining PINT-services based on the special client-server architecture and protocols [9].

Another VoIP approach by IETF is being undertaken within the SIP (*Session Initiation Protocol*) working group [10]. Its objective is to enable the call session management from WWW browsers. The SIP framework simplifies the call control (such as ITU-T Q.931 and H.235 specifications) and session management to become lightweight and WWW compliant. Its protocol is based on the traditional URL-notation (e.g. sip://+999-1-234567@net2ph.com) and it interoperates with traditional call control protocols.

To overcome the disparate and proprietary VoIP implementations companies and organisations have already launched projects to harmonise and uniform VoIP architectures and protocols. ETSI has established project TIPHON (*Telecommunications Internet Protocol Harmonisation Over Networks*) [11] to consolidate the different VoIP emphases. Its objective is to support the market for inter-operator voice communications and related voiceband communications (such as facsimile) between users. Some major vendors have further utilised the TIPHON work to define the OSP (*Open Settlement Protocol*). The objective of the OSP is to make operator-specific VoIP networks interoperate.

IETF is, on its behalf, currently specifying MGCP (*Media Gateway Control Protocol*) standard which integrates existing VoIP and media

switching protocols such as SGCP (*Simple Gateway Control Protocol*), IPDCP (*IP Device Control Protocol*) and SIP to form universal framework for interoperable Internet telephony architectures [12].

2.2.3 Service Infrastructure Consolidation and Convergence Platforms

The latest and most radical definition for Network Intelligence has arisen from the need to realise multi-networked value-added telecommunications services. The objectives of these platforms are to consolidate the telecommunications and Internet infrastructures and technologies to form an open and universal service network environment, Network Intelligence. At present, this means basically consolidating fixed (IN, PSTN, cable TV), mobile (GSM, PCS [*Personal Communications Service*]) and Internet (WWW in particular) service environments to form universal and consistent platforms for service design, creation, control and management.

One of the first concepts to propose the aforementioned objectives was TINA (*Telecommunications Information Networking Architecture*), which literally re-defined the networking architecture and telecommunications service distribution. Although the TINA vision is state-of-the-art, it is utterly complex. Therefore, it is not anticipated that full TINA-based networks will be implemented, but instead, applicable parts of the visions and architectures will be included in next-generation networks. One of the most important TINA architectures is the DPE (*Distributed Processing Environment*) [13] which defines IP-based distributed service networking environment for next generation converged networks.

Yet a more awaited and important standardisation for the NI integration is UMTS (*Universal Mobile Telecommunications Service*). The UMTS vision will ideally bring about the integration of wideband and broadband ubiquitous mobile networking with Internet services. UMTS will focus on building on existing and emerging GSM and Internet standards thus being an evolutionary rather than revolutionary standard.

Fortunately, the mobile telecommunications-Internet networking integration is underway with the introduction of the GSM Phase 2+ GPRS architecture. Also, while another GSM Phase 2+ standard called CAMEL (*Customised Applications for Mobile Enhanced Logic*) [14] already defines the IN-GSM service interoperation (and hopefully leads to a fixed mobile service platform consolidation), GSM truly offers an attractive environment to implement fullblown wideband mobile-based NI functionality in the near future.

In addition, a generic approach to the networking consolidation is the *Network Intelligence Middleware* (NIM) [15] based technologies. Their objectives are seamless interoperation of disparate networks and systems, and it combines the NI paradigm with the paradigm of middleware technologies. The NIM technologies will at the same time be flexible towards fast changing business needs (services, charging, customer care), guarantee mission critical integration to production networks and systems, and control the services in a reliable environment.

3. CONCLUSIONS

The convergence is undoubtedly revolutionising the communications technologies and business, merging networks of different types and companies of different origins. The future of the telecom business depends on how well old solutions can be combined with new ones to serve new customer needs.

It is obvious that a new breed of technologies will emerge to solve the fundamental differences between telecommunications and Internet technologies, and to integrate the present and future online content services. The characteristics of these technologies are far beyond what has now been already deployed: they will be based on open platforms, they will be multi-interfaced, they will be distributed in nature, they will be componentised yet integrated, they will interoperate autonomously and spontaneously, and finally, they will be reliable and scalable performance-wise.

It is neither easy nor convenient to try to predict what the dominant designs of the converged era will be, but some trends can be observed as described in this paper. Clearly, the most important trend of all is the IP integration and the technologies which have emerged and will be based on the IP foundation such as WWW, Java, XML, WAP, GPRS, NIM and UMTS. This integration takes place mainly in the service and switching layers, and thus accelerates the adoption of leading edge data communications technologies such as gigabit ethernet, WDM, ATM and VoIP. Another trend is the overall integration of the access and transport network interfaces: there will be various different appliances and thus access networks, but they will be tightly integrated to each network operator's terabit core transport network.

At the same time it is not only the networks and services which are converging. Appliances from different vendors are increasingly multifeatured including voice and productivity applications (scheduling, address book, word processing, spreadsheet, WWW browsers, etc), wireless data connectivity and other sophisticated capabilities. It is clear that there

will be a further consolidation in the technologies in this area. Besides these technologies, there will be the need for new business models and earning logics for the next generation converged businesses. In future, the present customer-supplier relationship and value chains will be totally redefined as has happened in Internet based commerce.

REFERENCES

- [1] Hurley, Hanna, *See JAIN Run*, Internet Telephony Feb' 1, 1999.
<http://www.internettelephony.com/archive/2.1.99/isnews.htm>
- [2] Parlay Homepage, <http://www.parlay.org/default.htm>, January 1999
- [3] Draft UMTS 23.01 v0.1.0, *Universal Mobile Telecommunications System (UMTS)*; General UMTS Architecture, September 1997.
- [4] Eurescom, *CORBA as an Enabling Factor for Migration from IN to TINA: A EURESCOM-P508 Perspective*. OMG telecom domain taskforce papers. 1998.
<ftp://ftp.omg.org/pub/docs/telecom/97-01-01.pdf>
- [5] Wyckoff, McLaughry, Lehman and D. A. Ford, *T Spaces*,
<http://www.almaden.ibm.com/journal/sj/373/wyckoff.html>, April 1998
- [6] Jini Connection Technology Homepage, <http://www.sun.com/jini/>, May 1999
- [7] IETF VoIP Group Homepage, <http://www.ietf.org/voip/>, August 1998
- [8] IETF PINT Working Group Homepage,
<http://www.bell-labs.com/mailling-lists/pint/>, August 1998
- [9] Faynberg and Lu, *Minutes of the PSTN/Internet Interworking (pint) BOF Meeting*, April 1997
- [10] Handley, Schulzrinne, Schooler and Rosenberg, *Internet Draft SIP: Session Initiation Protocol*,
<http://www.cs.caltech.edu/~schooler/papers/draft-ietf-mmusic-sip-12.txt>, January 1999
- [11] Tiphon Homepage, <http://www.etsi.org/tiphon/>, April 1998
- [12] Schulzrinne and Rosenberg, *The IETF Internet Telephony Architecture and Protocols*, IEEE Network May/June 1999
- [13] Chapman, *Overall Concepts and Principles of TINA*,
<http://www.tinac.com/specifications/documents/overall.pdf>, February 1995
- [14] DTR/SMG-032320U v0.0.3, *Evolution of the GSM platform towards UMTS*, September 1997.
- [15] Intellitel Communications Ltd, *Network Intelligence Middleware White Paper*
<http://www.intellitel.com/whitepaper>, April 1999

THE WANDERING LOGIC OF INTELLIGENCE

Or Yet Another View on Nomadic Communications

Plamen L. Simeonov

Siemens AG – Berlin, Information and Communication Networks, Communications on Air, CTO

Key words: Intelligent Networks, Service Architecture, Network Convergence, Nomadic Service Logic.

Whatever is flexible and flowing will tend to grow ... (Dao De Jing)

Abstract: This paper presents a third generation (UMTS/IMT2000) integrated network architecture among Flexible Intelligent Network Elements (FINEs) based on the multimedia multimode (client, server and agent) IN Service Node concept ([1],[2]), to perform a crafty Unified Media Communications Service (UMCS) across heterogeneous networks. The FINE architecture represents a network of configurable elements allowing user access to a Unified Message Store (UMS) and Universal Communications Channel (UCC) via traditional PSTN/ISDN/PLMN equipment such as telephones, pagers and fax machines on the one side and networked computers and mobile terminals equipped with mail readers, web browsers and video codecs to enable both on-line and off-line interactions. A virtual cluster of FINEs within an integrated personal communications network is dynamically configured in a distributed or centralised manner according to user profile requirements, network size and performance to provide artful messaging and conferencing services in a changing environment. Each FINEC can be deployed in some of the three modes --- independent (server), dependent (client) and autonomous (agent) --- with respect to the user/network configuration, thus allowing a dynamically configurable (per user/ per node/ per service) centralised or distributed service architecture. The FINE itself is organised along with other nodes by a Network Operation Support Environment (NOSE), a service oriented and TMN compliant Operation, Administration & Maintenance Center. Service logic can be transferred, installed and mounted on demand among the FINE Controllers and the terminal equipment to provide optimal QoS.

1. INTRODUCTION

The core of the proposed new IN architecture is the *FINE*, the result of a joint project initiative, *@INGate*, between Siemens AG and the Technical University Berlin, [1]. The goal of this project was to design an InterWorking Unit (IWU) as a mediation device between existing circuit and packet switching networks by using traditional ISDN/PSTN/PLMN equipment (TE2) and Internet Terminals (IT), cf. Fig. 1.

Our prototype hosts a generic Unified Media Communication Service (UMCS) composed of multimedia messaging (store-and-forward), telephony and agent-based network computing.

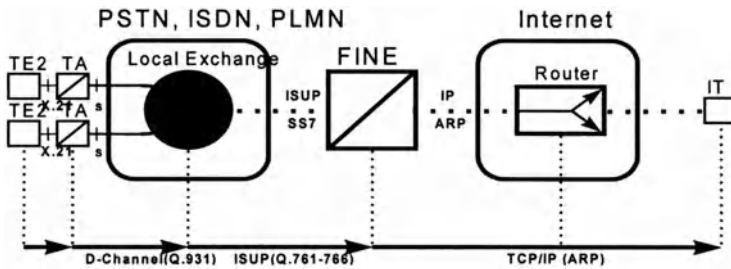


Figure 1. The Network-Bridge FINE as an InterWorking Unit

2. THE REFERENCE MODEL

Our Reference Model contains the following network elements (Figure 2):

- Flexible Intelligent Network Element (FINE)
- FINE Data Server (FineDS)
- Network Operation Support Environment (NOSE)
- WWW Server and Firewall

The Network-Bridge FINE architecture represents an InterWorking Unit between PSTN/ ISDN/ PLMN and the Internet. It is based on the Service Node concept in Intelligent Networks that was extended by cunning media treatment and routing techniques along with enhanced SRF, FSL and scheduling mechanisms.

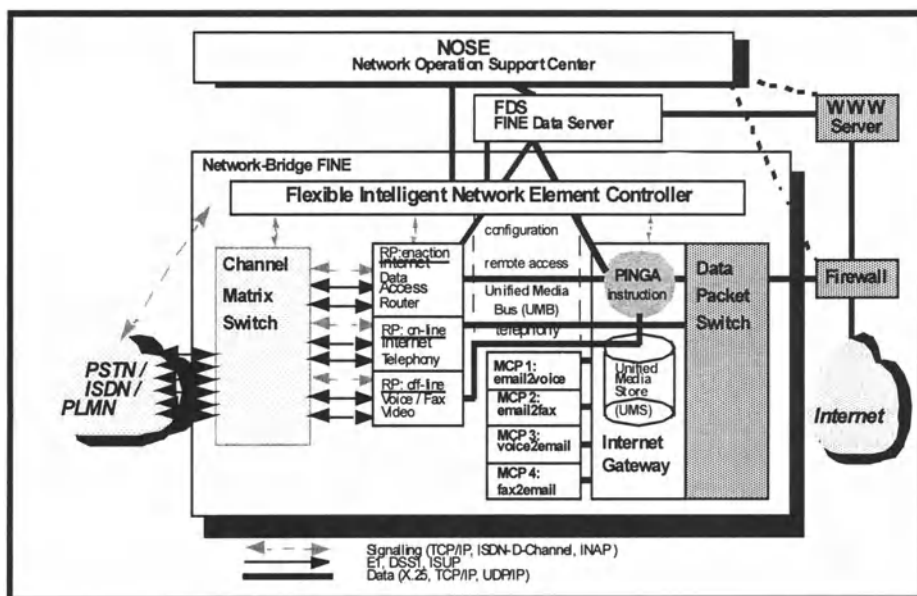


Figure 2. The @INGate Reference Model

The FINE consists of a FINE Controller (FINEC), one or more Resource Platforms (RPs), a Channel Matrix Switch (CMS), a Data Packet Switch (DPS), Media Conversion Processors (MCPs) and an Internet Gateway (IG) with a Unified Media Store/Bus (UMS/B), and a Personal IN Gateway Area (PINGA).

All FINE components except the FINE Controller which carries multiple service logic are optional and configurable to allow a *multimode* functionality of each single element.

The FINE Data Server (FineDS) holds all service and subscriber relevant information such as service configuration and user profile records.

The Network Operation Support Environment allows access to all FINE components (and of the neighbor FINEs) via remote control sessions to UNIX/WWW hosts (FINECs) or via SNMP-based management.

The *Firewall* computer connects the FINE to the Internet.

The WWW Server supports WWW-based management of all FINE components in addition to the TMN-conform NOSE workstation, as well as Customer Service Control (CSC) such as configuration of service specific parameters through the Internet by using security-enhanced Web browsers. Please refer to [2] for details about the model.

3. THE COLLISION OF THE NETWORK PARADIGMS

There is a definite distribution of functions and roles among network components within outbound-signaling telephony based IN. Thus, the standard SCP-IP relationship is e.g. not simply the client-server one (typical for data networks), but rather a processor-coprocessor delegation of service logic (e.g. playing announcements and collecting user input) to a subordinate unit, the Intelligent Peripheral, which delivers back some result to the Service Control Point. The SCP then continues to control the flow of the service in the well-known *main()* routine manner. Even in a distributed IN, Service Nodes are not supposed to serve each other in the common "data packet" sense, but rather to have a local (centralized) control over certain resources, services and subscribers. In addition, ongoing IN standardization do not consider alternative network technology integration paradigms, but rather extend the SS7 model by ROSE functionality, such as e.g. invoking functions (cf. ETSI INAP CS-2).

On the other hand, the packet data world is handling intelligence not in a call flow but rather in an application oriented way while addressing *variable* bandwidth technology, resource reservation (RSVP, [15]) and OSI protocol *stack permutations* such as e.g. running SS7 over TCP/IP ([6]).

Such developments are a clear sign that the layered architectures and the *fixed* treatment of IN functions, databases and services can no longer exist in a heterogeneous distributed network under rapid development where even *sculptured object paradigms* such as CORBA, TINA, or DCOM and powerful IN architectures based on "clustered, multiprotocol SCPs" (Figure 3) cannot handle all aspects of the network evolution as far as they remain *rigid*. One major aspect of the network has been neglected until now: it is *temporal* and *living*, i.e. *changing*. For example proprietary IPs can grow up to autonomous multiprotocol "virus-like" CTI point solutions hiding micro data networks in themselves.

A time for change in network design thinking has come. Now, we have the chance to adopt a *stepwise approach* for building the kernel of a new communications cell. Where could we start from? Perhaps *flexible nomadic intelligence* may be an option.

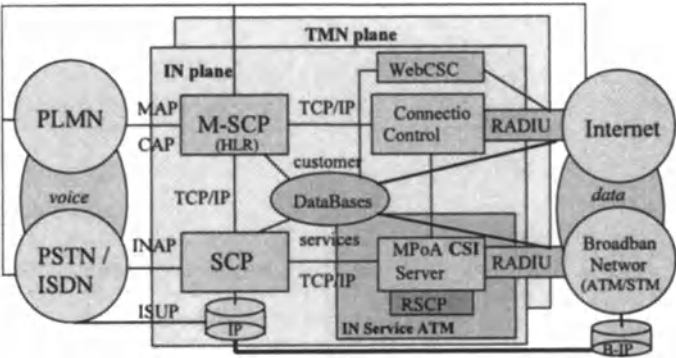


Figure 3. A Clustered SCP Configuration for Converged Networks

Solution Approach: The FINE Controller is the heart of the FINE node. It handles the allocation and deallocation of resources and the overall local logic of the service. The basic concept behind a *vivid nomadic network intelligence* is that each FINEC can be deployed in some of the three modes --- independent (*server*), dependent (*client*) and autonomous (*agent*) --- with respect to the user/network configuration.

This model allows a dynamically centralised or distributed service architecture. Services are implemented in the FINEC by *true* Flexible Service Logic (FSL) components interfacing to their subparts, micro service logic (μ SL, μ -services), and resources in the resource platforms (RP) directly or via interfaces to other FINEs in the network according to the ‘multiple SCP-IP’ model, cf. Figure 4.

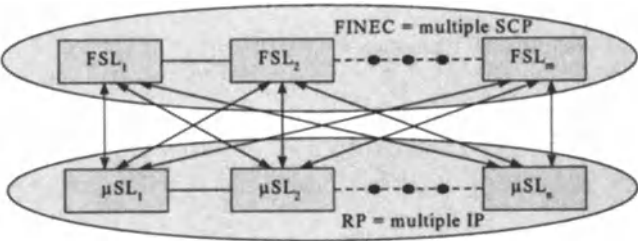


Figure 4. Distribution of Service Logic between SCP and IP

In the following, I will show how flexible intelligent network elements can be used for functionally integrated multimode logic control in advanced *nomadic intelligence* architectures.

4. THE FINE ARCHITECTURE

Figure 5 below illustrates the idea of a configurable network node that is essential for the FINE architecture.

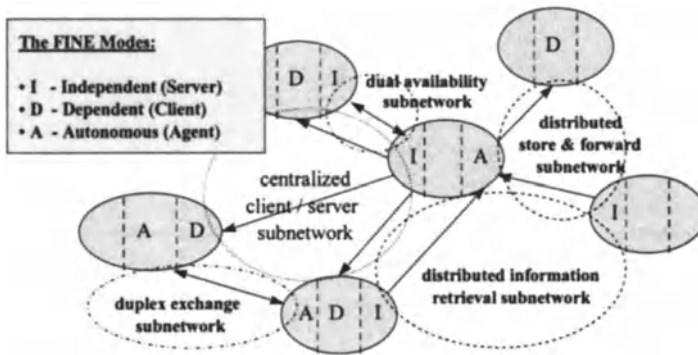


Figure 5. An Example of a FINE Configuration

Definition 1: A *Flexible Intelligent Network Element (FINE)* is a configurable multimode network element. It can be deployed in a single, dual or triple mode of operation as independent (I), dependent (D) and autonomous (A) unit, i.e. as *server*, *client* or *agent* accordingly. All three modes can be deployed at the same time as far as this does not conflict with basic network design principles. A FINE may be a physical entity, a piece of software or a virtual subnetwork by itself.

Even network terminals can be regarded as FINEs operating in some of the three modes.

Definition 2: The *FINE Architecture* is a versatile network intelligence construction of FINEs which are *temporarily* adopting certain roles in the network. It is required that

- agents be assigned to negotiate about the distribution of roles among FINEs (active network), and
- data, functions (service logic) and roles (modes) migrate upon request from one FINE node to another, thus allowing for dynamic network configuration.

This simple model offers a unified structured and evolutionary approach to intelligent network design and configuration which addresses the two basic architectures, centralized and distributed IN and allows for a flexible, but *deterministic* (event-driven) *spread of intelligence* across the network matching user demands wherever and whenever required.

This approach allows for effective treatment of problems such as feature interaction, service mobility (CAMEL+), database updates, etc.

The FINE model can be easily implemented using approaches such as JTAPI [4] for applications, JINI [14] for services and devices, or ATP (Agent Transfer Protocol) [5] for transferring service logic as mobile agents between networked computers.

The main advantage of this approach is that it allows sophisticated network growth, adaptation and rapid introduction of new services by making only minor changes on the available infrastructure. In this way, the FINE architecture represents a *vivid object-oriented model* for intelligent service provisioning and control when compared to the traditional horizontal and vertical OSI-like layered IN architecture.

4.1 Example A: Centralized FINE Architecture

Let us have a centralised loop video-on-demand service architecture (cf. [16], Figure 6) where media contents are transferred from a Central Media Server (CMS) to terminals throughout the network. Signalling and control are performed centrally (inbound or outbound) using the SCP-IP dialogue scheme. Media transfer is performed in a continuous synchronous channel mode or in a discontinuous asynchronous buffering mode. The CMS system provides media transfer, control and management from a single site. If the system fails or becomes incapable to support connections, connections are dropped unless there is a backup CMS.

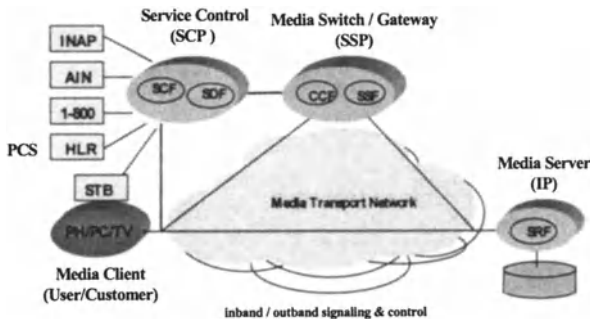


Figure 6. Centralized FINE Architecture

Here the FINE architecture can be configured to represent a centralised IN architecture. It requires three network elements: a media server (I mode), a media terminal client (D mode) and a control point server/client (I/D modes).

A high availability architecture as a second SCP and additional CMS can be modelled by introducing two server/client/agent (I/D/A modes) and two server/agent (I/A modes) elements instead of the single server/client and the single server respectively.

4.2 Example B: Distributed FINE Architecture

Let us now have the same video-on-demand service as a distributed multi-server architecture that spreads out the CMS functionality throughout the network using the concept of Local Media Server (LMS). This configuration saves communication costs and increases system reliability at the expense of local storage capacity by allowing subscribers to access media directly via Local Distribution Switches (LDS).

The central servers are located at the Media Information Provider (MIP) sites (Figure 7). Each LMS may be a mini-CMS and its media contents may be downloaded off-line from some CMS and updated periodically.

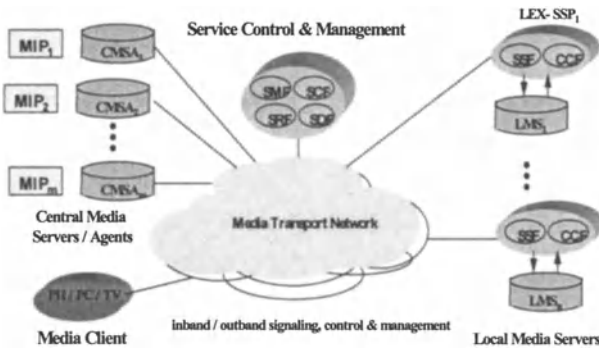


Figure 7. Distributed FINE Architecture

Here FINE may be configured in several variants from a cluster of purely centralized subnetworks of m CMS sharing n LMS to a cluster of purely distributed subnetworks of n LMS. In a distributed subnetwork, an LMS can be represented as a server (I mode) and a media terminal as a client (D mode). Modeling high availability is similar to the centralized case.

Furthermore, a single FINE may become an agent (A mode) to match SCP/SMP functionality by identifying LMS locations within distributed subnetworks and managing the overall service within the macro-network.

Of course, this function can be also cloned into a cluster of managing agents, one for each subnetwork. Then, why should not we keep this one and other options free for choice and configure the *network on demand*?

5. NOMADIC SERVICES AND THEIR LOGIC

The idea of migrating service functionality comes out of traditional network convergence solutions such as the One Number Service (ONS) for fixed and mobile networks. Figures 8 and 9 illustrate two possible variants to realize this service architecture.

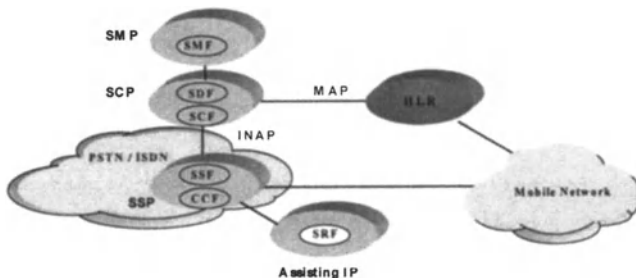


Figure 8. One Number Service for PLMN and PSTN based on Assisting IP[§]

The essential difference between them is the location of the Service Resource Function (SRF). We can simply ask what would come out to being if we *let this function roam* between the network elements for the reasons of evolution, effective resource utilization, adaptability and QoS guarantee.

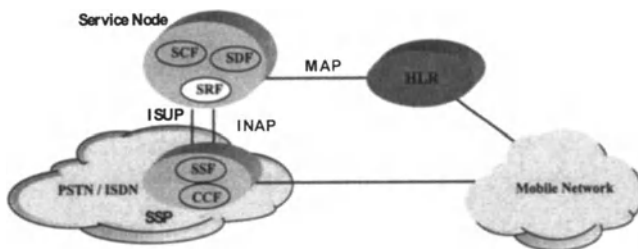


Figure 9. One Number Service based on Service Node with Integrated SRF[§]

To assist new services and customers, intelligent networks require permanent upgrades in hardware and software components which are not always optimally exploited at their location (Figure 10). Even a distributed architecture which dedicates certain resources for local usage is not adapting (not yet) to variable load and service usage.

§ - with the kindly permission of Dr. R. Rieken, Siemens AG: IN'97 Workshop (IEEE), May 1997, Colorado, USA

SCF + SDF	:	Number Translation, VPN, ...
SCP + SDP + SRF	:	Voice Recognition, UMS, ...
SCP + SDP + SRF + SSF	:	Local Operators, ...
SCP + SDP + SRF + SSF + GWF	:	Internet Telephony ...

Figure 10. Overall Trend - Increasing Enhancement of IN Functions

Thus, next to *HOW*, the question of *WHERE* to physically implement an IN function has always been crucial for network designers to provide for more efficient and effective services. Now, in the era of platform independent OO technology this subject obtains a new thread: *mobility*.

Therefore, we still have two choices:

- a non-deterministic worst-case-dimensioned service within a rigid network architecture requiring evolutionary software and hardware upgrades, or
- a deterministic service within a dynamically configurable network architecture, *FINE*, based on autonomous nomadic services taking care for software updates and optimal utilization of hardware resources to a certain grade that can be proved by algorithms.

In some cases, it seems reasonable to temporarily enact and allocate/move different services and their resource areas depending on the eventual customer demands, instead of permanently increasing the functionality and the capacity of the network elements (Figure 11).

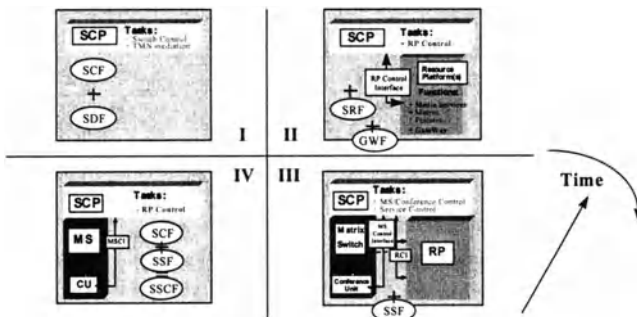


Figure 11. A Possible FINE Node Utilization Cycle

This implies an event-driven, i.e. deterministic, redistribution of roles among the network elements (i.p., client, server and agent) to satisfy network performance and user demands on configuration and QoS (cf. Figure 12).

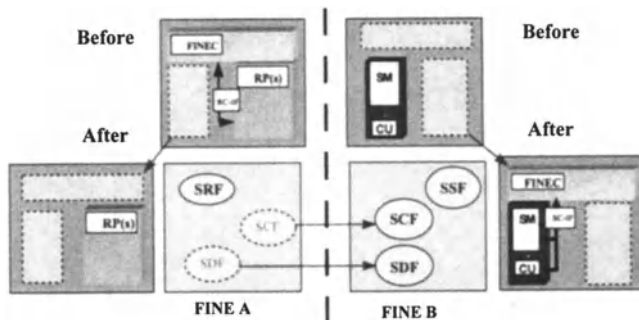


Figure 12. A FINE Reconfiguration via Function Split, Migration and Clustering

Definition 3: *Nomadic FSLs* (flexible service logic) are the smallest autonomous service elements known as features (functions) including parameters (variables) and database contents being transferable from one (fixed or mobile) node of the network to another to offer *intelligence on demand*.

Thus, the FINE model leads to an adaptable, learning, self-configurable service intelligence architecture where the database and feature mobility are parts of the service itself. Although the FINE approach has an IN telephony origin, it can be used to virtually (re-)configure in a dynamic manner *any type of network*.

6. THE WANDERING LOGIC OF INTELLIGENCE

Recent research on active networks approaches the problem of effective service provisioning not by moving services along network nodes, but rather by using the means of (mainly Java driven) object-oriented encapsulated software technology up to the deepest layers and elements of network protocols. No matter of whether we talk about a *Netscript* to program functions for sophisticated packet stream processing in network nodes [7], evolving to form *capsules* or datagrams carrying code [8], *sprokets* or “smart packets” [9], *switchlets* or “active packets” [10], *CANEs* or “composable active network elements” [11], -- in all these works the search for data and process customisation beyond the limits of performance, up to the application, up to the user needs is evident.

The belief creates the reality. This holds until the next change of the paradigm. Yet, we still miss the quantum jump, the profound uncertainty equation that tells us where are the limits of networking management.

For the time being, there are basically two approaches to improve an application's performance: a service oriented and a network oriented, both considered as *objects*, both strictly separated in hardware and software.

Programming is mainly a *soft* method that leaves certain options and degrees of freedom to adjust a model's behaviour towards a desired result. If a high-level abstraction program performs well, it is then translated into a "lower" language and eventually split into modules and instructions that fit the hardware and thus better perform. Finally, if this works well, the entire program is "burned" into silicon to match the "click-to-switch" or "be there" requirement and perform for the best. Further improvements are only due to the physics.

Unfortunately, there is no way back ... to check if the things could work the other way up.

If you have a new idea, you need a new model that most probably needs a new language to express its axioms, predicates etc. logic constructs. The trouble with modern computer science, however, is that it still lacks the freedom of the greek philosophy era. Every problem is digitalized and reduced to a set of objects and functions (both *corpuscles*) that express the same formalism in the broad sense. Be in hardware or software, it is the same logic and the same *programming* language, no matter whether we unfold services or inject pro-active capsules into network nodes.

What we probably miss is the analogous *wave* in the digital world, the sparking *process* that describes the quantum nature of evolving multiple realities, some kind of a free, *Wandering Logic of Intelligence (WLI)* that can be at least welcome by the Lamport's strong fairness formula [13]:

$$SF(WLI) = (\Box \Diamond \langle WLI \rangle_f) \vee (\Diamond \Box \neg Enabled \langle WLI \rangle_f)$$

where $\langle WLI \rangle_f = WLI \wedge (f' \neq f)$, and f is its state function.

7. CONCLUSIONS

The FINE architecture presents a new service view at IN (in ODP terms) when compared to the classical vertical and horizontal layering of intelligence in networks. It can be centralized, distributed or both at the same time depending on the network size, architecture, service configuration, performance, scheduling and QoS requirements, as well as on the subscriber and service migration flow.

This paradigm can be used as a *unified, vivid object-oriented IN* model where the roles of the network elements (SSP, SCP, SMP, SN, IP, servers, routers, CPE, etc.) are temporary and on demand.

Within the FINE concept of a dynamically configurable intelligent network, services and their logic are gaining a new value by becoming

autonomous and truly mobile entities throughout the nodes of the network reaching even the terminals to turn them temporarily into agents, clients or servers whenever and whenever required. Thus, service logic can be transferred, installed and mounted on demand among the FINE Controllers and the terminal equipment to provide optimal resource utilisation and QoS.

The FINE Architecture Advantages

Replacing the Deterministic with

1. **Undeterministic Approach to Service Deployment**
according to Actual User Requests and QoS Requirements

Replacing the Fixed Distribution of Roles within the IN and Data Networks
incl. Tunneling (Encapsulation) of Communications with

2. **Flexible Distribution of Roles and Functions** (SCF, SDF, SRF, GWF, etc.)
3. **Dynamic Network (Re-)Configuration** on User Request involving

- *Temporary Load Distribution*
- *Temporary Resource Sharing, Control & Distribution*
- *Variable Channel and Bandwidth Allocation*
- *Reuse of Software and Equipment*

The FINE architecture is still under discussion and refinement. Currently, Siemens is offering a Messaging Gateway as a front-end solution to the SCP (INXpress 6.1) for multimedia services at the boundary to Internet. Yet, this function *may move* to other network elements in future.

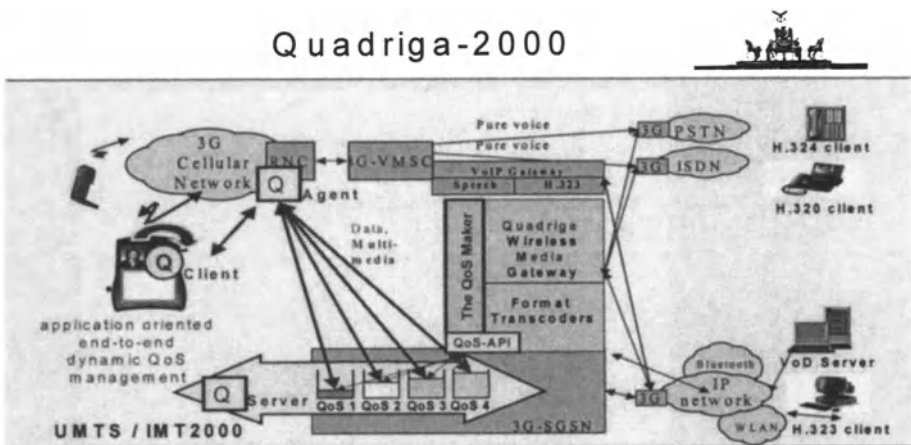


Figure 13. The QoS Maker Management Architecture for 3G Wireless Video Telephony and VoD over IP

A typical example for applying the nomadic service logic approach is the *dynamic QoS management* (Figure 13).

Siemens is currently investigating this subject in an active network architecture for 3G *wireless* multimedia delivery with the Quadriga-2000 initiative. For more details about this project please refer to the author.

8. ACKNOWLEDGEMENT

The author would like to thank Prof. Dr. Dietrich Reschke from the Technical University of Ilmenau and Dr. Andrea Heilingbrunner from Siemens AG, Munich for the support that made this work appear.

REFERENCES

- [1] Plamen L. Simeonov, *Add-On IN Gateway Services for Service Node in Distributed Telecom Networks*, Nr. P30308-A8720-A000-04-7618, Siemens AG, Oct. 1996.
- [2] Plamen L. Simeonov et al., *A Smart Node Architecture Linking Telephony with the Internet*, Proc. of ICC'97, Cannes, Nov. 18-21, 1997, 77-81.
- [3] Tsong-Ho Wu et al., *Distributed Interactive Video System Design and Analysis*, IEEE Comm. Mag., March 1997, 100-108.
- [4] <http://java.sun.com/products/jtapi/index.html>.
- [5] <http://www.trl.ibm.co.jp/aglets/atp/atp.htm>.
- [6] <http://www.ttiweb.com/corporate/press/ssip-pr.htm>
- [7] Y. Yemini and S. daSilva, *Towards Programmable Networks*, IFIP/IEEE Intl. Workshop on Distributed Systems: Operation and management, Oct. 1996.
- [8] D. Tennenhouse et al., *A Survey of Active Network Research*, IEEE Commun. Mag., Jan. 1997.
- [9] B. Swartz et al., *Smart Packets for Active Networks*, <http://www.net-tech.bbn.com/smtpkts/smart.ps.gz>, Jan. 1998.
- [10] D. Alexander et al., *The SwitchWare Active Network Architecture*, IEEE Network, Special Issue on Active and Programmable Networks, May/June 1998.
- [11] E. Zegura, *CANes: Composable Active Network Elements*, Georgia Tech., <http://www.cc.gatech.edu/projects/canes/>.
- [12] D. S. Decasper et al., *A Scalable High-Performance Active Network Node*, IEEE Network, Jan/Feb 1999, 8-19.
- [13] L. Lamport, *The Temporal Logic of Actions*. SRC Res. Report 79, DEC, Sys. Res. Center, Dec. 1991.
- [14] *Jini Connection Technology Specification 1.1*, June 1999, http://www.sun.com/jini/specs/jini1_1spec.html
- [15] Zhang, L., Deering, S., Estrin, D., Shenker, S., and Zappala, D., *RSVP: A New Resource ReSerVation Protocol*, IEEE Network, September 1993.
- [16] Tsong-Ho Wu et al., *Distributed Interactive Video System Design and Analysis*, IEEE Comm. Mag., March 1997, 100-108.

PLUG-AND-PLAY NETWORK SERVICE CONFIGURATION USING CORBA

Syed Kamran Raza, Bernard Pagurek, Tony White

*Dept. of Systems & Computer Engineering, Carleton University, Ottawa, Canada
skraza@nortelnetworks.com, {bernie, tony}@sce.carleton.ca*

Key words: Plug-and-Play, Service Provisioning, Configuration Management, CORBA

Abstract: Configuration of network services is a difficult problem requiring considerable human involvement. Recently, the hardware configuration of a personal computer has been simplified by the Plug and Play specification and the addition of the universal serial bus. Also recent developments in the CORBA framework, the Java language, mobile agent facilities, and Jini, have provided new tools that facilitate the movement of code from device to device, and help to provide a framework for service subscription and utilisation on demand. This paper brings together the principles of the Plug and Play specification, CORBA, and mobile code in a design for plug and play network services that should provide more interoperability and language independence than the Java-centric approach

1. INTRODUCTION

Configuration of network services is a difficult problem, which requires considerable human involvement. As the installation and configuration of a PC peripheral was made easier through the introduction of Microsoft's and Intel's jointly proposed plug-and-play (PnP) approach [3], [4], the network service configuration problem can also be facilitated by adopting a plug-and-play approach.

The concept of PnP from the PC world, however, requires an acknowledgement of added complexity seen in networks because there is no centralised OS available in networks that could handle the auto-installation

scenario. Furthermore, maintaining a registry (like Windows) of all the network components/services with their interaction schemes is almost impossible. Finally, provisioning a network service is usually more complicated than PC driver installation.

Technologies are only now becoming available that could be used to support a plug-and-play capability in networks. For example, CORBA [6] has introduced naming and trader features [5] that provide for service discovery, thus allowing potential clients of a service to locate a service entity. Furthermore, CORBA 3.0 [13] offers a form of code mobility through exploitation of passing object-by-value that would facilitate the deployment of client-side code on the client device thus enabling service utilisation.

The research reported in this paper is part of an overall effort to evaluate recent emergent technologies and their use in the implementation of plug-and-play services. Our previous work on plug-and-play network services has included the evaluation of a Jini-based implementation [10] and mobile agents [7]. Further, [9] presents an evaluation of these distributed computing technologies for network service configuration. A CORBA evaluation is important in that mobile agents and Jini [11] represent (largely) Java-centric solutions; interoperability thus suffers. A CORBA approach therefore promises to provide language independence and interoperability with legacy components.

2. MOTIVATIONS AND OBJECTIVES

The services over a typical network are usually installed on a server and shared among clients. The servers usually provide access to some resource on the network such as a printer, storage disk, or router. Often, there is a need for installation of corresponding software on the client machine for that resource. We term these resources and their associated software as *service* and *service module* respectively.

The focus of this paper is the Network Management problem of *configuration and related software distribution* that arises with the installation of a new service provider component. On a broader scale, our objective is:

- To make service provision easier to set up and configure in networks.
- To provide continuous supervision of the networks for service provisioning due to networks' dynamic nature.
- To minimise the interactions between a user and a network component during its installation.

- To minimise network administrator intervention in a component's configuration.

We propose a Plug-and-Play framework for networks that is composed of smart, intelligent service provider components. After some manual steps that are required to install these components in the network, the components would make their installation and configuration arrangements themselves without user intervention. We associate the term *Plug-and-Play component (PnP)* with such components. Although, a network printer is chosen to represent a service provider component for describing the network PnP framework in forthcoming sections, more general plug-and-play services can easily be accommodated within the framework presented here. Classical object-oriented analysis and design methodology has been adopted for defining our PnP system.

The system of our concern is the framework that provides plug-and-play capability and its environment includes the actors that use or affect this system via interfaces. On inspection, we discover five major actors that interact with our framework -- *Vendor*, *Network Administrator*, *Service provider network component*, *Client network component*, and *Broker*.

As stated earlier, a network component that offers some kind of service (s) to the other network components is called a Service Provider component while one that requires or uses an offered service is called a Client component. For example, a network printer is a service provider while all other workstations that wish to use the printer are its clients. The vendor is the party that supplies a service provider component and its associated software. The Network Administrator is the person who is in charge of managing the network. Finally, the broker serves as glue between clients and server components and participant components find each other using this broker.

3. REQUIREMENTS AND FACILITIES

3.1 Requirements

The PnP framework is the system that is required to provide the following functionality and capabilities in the components:

- Auto installation of client and server components
- Discovery-and-Join of the network by participant components
- Service provisioning by PnP component
- Service discovery and utilisation by client components
- Service upgrades

- Dynamic configuration provision

However, installation and configuration cannot be automated completely; there are tasks that require user intervention. Examples of these tasks include establishment of a physical network connection, supply of initialisation information related to the configuration, set up and security, and monitoring of system messages, error messages and logs, etc.

After determining the major requirements of our system, we organised the requirements by assigning appropriate responsibilities and functionality to each actor [8]. The assignments also led us to discover essential interactions among components.

3.2 Facilities

After performing more analysis, we also determined various facilities that are required to architect our PnP system. These major facilities, in summary, are Naming/Trading, Security, Notification (events), Persistence (storage), Transaction, and Logging.

3.3 Component Interfaces

The framework consists of many components that need to know each other's interfaces to collaborate for achieving the PnP functionality. Therefore, we also defined necessary interfaces (*PnP*, *Client*, and *Vendor* interface) for the main participants. A configuration interface is also defined, which is used by an administrator to configure the component. A component may implement more than one interface. For example, a service provider PnP component may also function as a client component for other services.

4. CORBA OVERVIEW

To solve the problems of application interoperability and portability with increasing heterogeneity in today's networks, the Object Management Group (OMG) has specified a distributed object framework, called the Common Object Request Broker Architecture (CORBA) [6]. The key feature of CORBA is its cross-platform communication architecture, which enables definition, transportation, implementation, and invocation of objects in virtually any programming language. All these features make CORBA a powerful enabling technology for network management purposes.

CORBA specifies various common object services [5] that are very useful for distributed computing. Further, those aspects of CORBA that promote interoperability are also quite important.

A high-level view of the CORBA architecture provides three main components -- a client, an object implementation (server), and an ORB. The *ORB* is the middleware component that acts as a broker and glues clients and servers together. The services provided by an object are expressed in the form of an *interface*, which is defined using the Interface Definition Language (IDL). CORBA defines two modes for the invocation of requests - static and dynamic invocation. Static invocation uses statically defined IDL information, compiled into two parts -- a client side *stub* and a server side *skeleton*. Some applications, for which priori knowledge of IDL (stub/skeleton) is not possible, require a dynamic invocation approach to interact with other applications. Furthermore, it is desirable to discover and use the new objects as soon as they are added to any ORB. CORBA provides these features through its Dynamic Invocation Interface (DII) and Dynamic Skeleton Interface (DSI).

5. CORBA BASED PLUG-AND-PLAY APPROACH

The CORBA-based PnP approach is a distributed application that realizes PnP participant components as CORBA objects. These components are distributed over CORBA-enabled networks, where an ORB acts as the underlying communication structure and middleware component, thus providing appropriate transparencies. Essential CORBA object services (Naming/Trading, Security, Events, etc.) and other ORB elements (Interface Repository, Implementation Repository, etc.) are also provided on ORBs.

5.1 Main Building Blocks

Figure 1 illustrates the deployment of the CORBA-based PnP approach. Before describing the PnP scheme, we describe the major building blocks.

Domains and ORBs

The PnP application is divided into two domains -- the vendor domain that comprises the vendor site and the vendor component, and the enterprise-domain that involves the network of interest, running various clients and server (PnP) network components. Each of these domains has an ORB on which the respective components reside. These ORBs not only bind local components but also inter-operate and communicate using the Internet Inter-ORB Protocol (IIOP) for enabling remote bindings.

Participants as CORBA Objects

Every hardware component (PnP and other network components) has appropriate ORB functionality pre-installed to function as a CORBA object and has a unique Object Reference (OR), which is assigned by its host ORB.

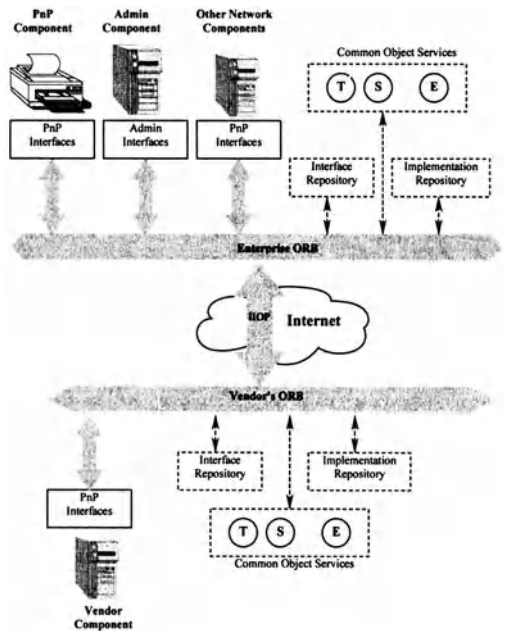


Figure 1. CORBA based PnP Approach

The underlying ORB and its directory service play the role of a broker actor.

Interfaces

For objects to ‘plug-and-play’ together in a useful way, clients must know exactly what they can expect from other objects. Therefore, the interface definitions are used to define the required contract/agreement among the clients and server objects. The OMG-IDL can be used to write the PnP interfaces in CORBA. An example PnP interface, written in IDL, is shown in Figure 2. This interface is implemented by a vendor component, helping the PnP components in (de) registering with their vendors.

```
module PnP {  
    struct ComponentInfo {  
        string name;  
        string owner;  
        string address;  
        ....  
    };  
  
    interface vendor {  
        readonly attribute string vendorName;  
        .....  
        boolean register(in ComponentInfo pnp) raises SecurityException;  
        void deregister(in long long registrationID);  
    };  
};
```

Figure 2. Example PnP Interface (in IDL)

Interface and Implementation Repositories

Since all CORBA based invocations (static or dynamic) require IDL interfaces, it is required that both the vendor's and enterprise's ORB maintain an interface repository, which is accessible by other ORBs as well. An Implementation Repository is also required on both the described domains for storing and retrieving the object implementations. This repository also provides other features such as the activation and deactivation of objects and their servers.

5.2 Useful CORBA Services

This PnP application benefits heavily from CORBA's object services. Although CORBA specifications describe 16 common services [5] that are used (directly or indirectly) by most of the CORBA applications, this section highlights only those services that provide a key functionality to this PnP application.

Naming and Trading service

CORBA provides two directory services -- Naming and Trading – that can be used by the clients for locating the objects across the ORB or network. The Naming service provides a mapping from a name to an object reference but it does not fulfil all the requirements in dynamic environments. For instance, a client component looking for a printer service that may print colour and double sided at a speed of 10 pages/min requires a sophisticated directory service. To fulfil such requirements, CORBA provides a Trading service. The trading service not only serves the basic function of name resolution, it also allows server objects to register their service offers, and clients to find the required services.

It is required therefore, that both vendors and the enterprise network run a Trading service. The components first obtain the OR of the trading service through the ORB-interface. Using this OR, a PnP component advertises its service, while the client components discover the services they require.

Security service

CORBA provides a detailed specification for a Security service. The service specifies the mechanisms that provide end-to-end security in multi-vendor object networks through appropriate security interfaces for the authentication, access control, recording audit events, and security administration. These features satisfy our PnP security requirements.

Event service

The proposed PnP approach comprises several scenarios where components require the event and notification services for asynchronous communication. The CORBA framework provides an Event and Notification service that may be useful for this purpose. This event service is designed to

provide the ‘change notification’ -- meaning that a change in an object (supplier) generates an event that is propagated to all interested parties (consumers) using another CORBA object, called the *event channel*.

Property service

CORBA’s Property service provides mechanism for associating the properties (i.e., <name,value> tuples) dynamically with an object. This feature helps in supplying initial configuration parameters to components.

Dynamic Invocations

Although the static method invocation works well for most of the PnP negotiations, an alternative dynamic approach may be chosen in some cases. For example, the clients that have no knowledge of the interface of an object can use the DII, whereas the DSI can be used when we want our object implementation to be changed and upgraded without disturbing or affecting its clients.

5.3 Scheme Description: Step by Step

Before describing the interaction steps and scenarios of the PnP scheme, the scope of CORBA needs to be clarified as follows.

The CORBA framework is used only for facilitating the installation and configuration steps of a PnP component. Once the corresponding service modules are installed and activated on a client component, the client components use the required service directly over the network without any involvement of the CORBA framework. For example, the PnP service configuration for a printer uses the CORBA framework, its transport, and its services to arrange the printer device driver on the client components. Later on, the client components communicate directly with the printer component over the network through the installed printer driver for printing.

Although this PnP approach does not mention security explicitly, it is assumed that all the sensitive interactions use CORBA’s Security service.

Component’s layout

Consider a heterogeneous network, which is CORBA-enabled. The participant components are connected over a common ORB object bus and the ORB is running the CORBA object services that are required for this application. These services are accessible to every component through the standard interfaces and include Naming, Trading, Security, Event, Property, and Persistence. Moreover, the other necessary ORB elements like Interface Repository and Implementation Repository are also available to the components.

Since the responsibility of maintaining the network lies on the shoulder of a network administrator, an administration application is running on the

enterprise network (as well as on the vendor's) that the administrator may use to configure the network components.

PnP component's installation

Now assume that the administrator realises the need for a new service -- say printing -- on the network. The administrator purchases a new service provider PnP component (printer) that has a built-in support for that particular kind of ORB. After its manual installation, the PnP component's software module connects the component to the ORB and finds the object references for essential ORB elements. The component registers its service with the Trader, implementation with the Implementation Repository (using a POA), and interfaces with the Interface Repository.

Vendor's set up

The PnP component's vendor also maintains an ORB for its product support. This ORB runs the same facilities and services that are specified for the enterprise ORB. The central component on the vendor's ORB is the vendor component which implements standard interfaces and is the main access node for vendor-manufactured PnP components. A PnP component, however, requires knowledge of its vendor for accessing it. Therefore, the vendor hard-codes the IOR (Interoperable Object Reference) and stub of its server component in each of its manufactured devices so that they are able to contact the vendor after being installed. Due to the use of persistent IORs, a component's reference remains valid even after its location changes.

PnP component's registration with its vendor

After executing an initialisation routine and getting appropriate references of the required CORBA services, the newly added PnP component invokes the registration interface on the vendor component using the hard-coded information. This registration interface may be proprietary because both the participants belong to the same vendor. The request goes through the component's ORB and is forwarded to the vendor ORB, whereas the vendor component object implementation is contacted through the Implementation Repository and the registration is completed. After registration, the PnP component may also invoke a standard interface method to download the latest server-side module and upgrade itself.

PnP component service advertisement and client's requirements

After the successful registration, the PnP component is ready to offer its service to potential clients. There are two modes for service provisioning -- *Push* mode in which the PnP component initiates the service offer and discovers its clients, and *Pull* mode, where a service provider PnP component allows interested clients to discover its service. These two modes can be incorporated in a CORBA-based approach as follows.

Assuming the Push mode first, a question arises immediately: "how does a new PnP component know which clients are interested in its service?" It is

proposed that each interested client registers with the Trading service and specifies its requirement. Then the PnP component queries the Trading service and makes a list of all the objects (i.e., its potential clients) that have shown an interest in its service.

The Pull mode, however, is more complex. In this case, the service provider component advertises its service with the Trading service, whereas a client component queries this Trading service for finding a service. If the service is not found, clients need to wait until such a service registers with the trading service. The CORBA's Event service is used to provide a notification mechanism; i.e., a 'service availability' event is defined for each type of service, whose consumers are the client components and supplier is a PnP component that offers the associated service. When the PnP component generates the 'service availability' event after completing its initial configuration routine, the registered client components are notified through the standard interfaces. Since the clients now have discovered their required service provider component, they can invoke a standard registration interface on the PnP component. The PnP component waits for a reasonable amount of time (depending on network latency, size of network) after generating the service availability event before assuming that all the interested clients have been registered.

Whether using the first or second mode for service discovery and requirement, the PnP component has knowledge of its potential clients at this stage. The next step is to find the client's platform (i.e., machine, OS, etc.) so that an appropriate service module may be arranged for the client component. The PnP component invokes a method on each registered (or discovered) client to retrieve the client's information. Note that clients may also supply this information during their registration routine with the PnP component in first case, thereby saving an extra method call by the PnP component on each client. The PnP component now has the pertinent information about its client components, from which the component determines the requirements of its client and constructs a requirements list.

Service module distribution and activation on clients

At this stage, the clients are ready to receive a service module. The PnP component invokes a standard module interface on the vendor component and passes the list of client components (their IORs) along with their required module types. The PnP component may use CORBA's deferred synchronous invocation mode for this operation and hence does not block waiting for the response. The vendor component receives the request from the PnP component, gets the required service modules from its repository, and sends these service modules to the corresponding client component by invoking standard PnP interface calls. It is to be noted that what is required here is sending an actual executable service module to the client component

and not its reference. One of the shortcomings of the CORBA 2.0 framework is that it only supports the option of passing object-by-reference and not “by-value” in method invocations, so we need CORBA 3.0 [13] which incorporates this object by-value (OBV) passing feature. In cases such as printer provisioning, where the service requires just a client-side counterpart, the service module may just contain executable code, whereas for some other cases, it may contain extra information about the server component so as to serve as a proxy at the client. Ultimately, the client receives this service module (as a byte stream) and activates it on its machine. In this manner, all the interested clients are configured and the vendor component sends a response back to the PnP component. With this, the initial cycle of installation and configuration of the PnP component is completed.

Dynamic configuration

For keeping up with the changing network environment, the PnP components should also configure any client components that are added after this initial configuration cycle.

In Push mode, a PnP component should poll the Trader service from time to time, check all the client components that are interested in its service, and filter out the interested clients that have not yet been configured. Once all new clients are discovered, the module distribution and activation process is repeated for these non-configured components. An alternative solution would be to use the Event and Notification service to notify the PnP component on arrival of a new interested client at the Trader service.

Future configuration support is implemented differently for the Pull mode. In this case, a newly added client component registers with the Trading service and queries it for any PnP component that fulfils its requirements. If such a PnP component is found, the IOR of the PnP component is retrieved from the Trader service and then interface methods are invoked on the PnP component that would, in turn, call provisioning routines for the new client component. If no PnP component is found by querying the Trader service, the client uses an event registration mechanism with the PnP event channel to receive notifications on availability of such a PnP component.

Service module upgrade

A major requirement in our PnP approach relates to the upgrade of service modules. Since upgrades may arrive any time, an event notification mechanism is needed. A vendor notifies all the registered PnP components directly whenever a service module upgrade occurs. The notified PnP components, in turn, call back the service module interface on the vendor to retrieve the upgraded service module. If the module is a server-side module, the module is installed and upgraded at the server PnP component (using the DSI mechanism). Whereas, if the upgrade is related to the client-side, then

the concerned client components (configured previously) need to be upgraded. The PnP component may query all the affected client components for their interest in the upgrade by invoking a standard PnP interface. For all those clients that have shown interest in the upgrade, the PnP component repeats the steps of module distribution and installation.

6. EVALUATION

6.1 Advantages

The CORBA environment suggests an architecture that runs on every platform. Therefore, the CORBA based approach provides a portable, object oriented and inter-operable application that is hardware, operating system, network and programming language independent. This approach allows development of dynamic applications that can be easily assembled and modified on the fly by using DII, DSI, and Trader facilities. The Trader facilitates dynamic discovery and late binding to the services as new objects can easily be added to the system and operations can be invoked easily using DII. On the other hand, the provisioning of DSI allows modifications (construction) of a dynamic server object at run time. The event and notification service makes the configuration task easier for dynamic environments. The use of DII and DSI also provides benefits of a smaller memory foot print since they only require a single interface for all the invocations when compared to the static mode, which requires a separate stub and skeleton per interface. The CORBA framework saves a programmer from worrying about the basic services for distributed objects (directory, security, transactions, persistence, concurrency, event) by providing common object services. The OMA standardises the component's interfaces that enable development of a PnP environment. The clients see only the server object's interface and not its implementation, allowing the substitutability (easy upgrade) of the server implementation. The client and server components are highly de-coupled, which provides a great amount of flexibility. Furthermore, any legacy network management application can easily be ported to the CORBA environment by giving it an IDL interface and wrapping it inside a thin CORBA object. Finally, CORBA 3.0 specifications include the feature of object mobility (Object-By-Value) within or between ORBs that will provide the kind of flexibility that is often found in mobile agent frameworks.

6.2 Disadvantages

Since CORBA is a complex and expensive framework, the requirement of ORB support on each component may not be acceptable for smaller devices. CORBA is not the best solution for real-time embedded systems due to its large footprint and performance issues. Therefore, if an ORB is to be burnt into a chip, it has to shed much of its functionality. This raises a trade-off between simplicity and power. CORBA 3.0 specifications are also making an attempt to define a Minimal/Embedded CORBA. Though CORBA specifications cover many services, most of the vendor's ORBs support only the major and essential services. Dynamic invocations (DII and DSI) are complex mechanisms, not all ORBs support them, and they require more effort and coding on the part of the programmer. This approach still requires some expertise from the administrator for maintaining its ORB. Since the PnP interfaces should be standardised and supported by all components, a separate standards group is needed. The existence of proprietary protocols among certain CORBA elements (mostly at the server object side) makes code portability from one ORB to another questionable.

7 CONCLUSIONS

The proposed CORBA architecture has a considerable number of advantages when compared to the Java-centric solutions described in [7], [8] and [10] owing to its language independence and extensive standard specifications. While a security service has been defined for CORBA, it is complex and, as a result, few implementations have been forthcoming. Several of the other services required by a CORBA-based PnP system are also not widely implemented. The most important obstacle to deployment of architectures such as reported here is in the area of standardised interfaces. It should be stressed that an implementation of this architecture – in its most general form – would require the object-by-value message passing mechanism only available in version 3.0 of the CORBA specification.

The essential plug-and-play ideas reported here are being applied in the area of advanced services for IP telephony [12] using concepts of mobile agents and Jini. Future work will explore the CORBA based approach described here for these services.

ACKNOWLEDGEMENTS

The research reported in this paper was supported by Communications and Information Technology Ontario (CITO) and Natural Sciences and Engineering Research Council (NSERC), Canada.

REFERENCES

- [1] Bieszczad, A., White, T., and Pagurek, B., "*Mobile Agents for Network Management*"; IEEE Communications Surveys, Sep. 1998.
- [2] Bieszczad A., Raza S. K., Pagurek B., and White T.; "*Agent-based Schemes for Plug-And-Play Network Components*"; Proceedings of the International Workshop on Agents in Telecommunications Applications (IATA'98); Jul. 1998; Paris, France
- [3] Halfhill, T.R.; "*Transforming the PC: Plug and Play*"; Byte Magazine, Sep. 1994; URL: <http://www.byte.com/art/9409/sec7/art1.htm>.
- [4] Microsoft Corporation; "*Microsoft Windows and the Plug and Play Framework Architecture*"; Mar. 1994; available at URL: <http://www.microsoft.com/win32dev/base/pnp.htm>.
- [5] Object Management Group; "*CORBA services: Common Object Services Specification*"; Dec. 1998; available at URL: <http://www.omg.org/library/csindx.html>
- [6] Object Management Group; "*The Common Object Request Broker: Architecture and Specification*"; CORBA/IIOP 2.2, Feb. 1998; available at URL: <http://www.omg.org/library/c2indx.html>.
- [7] Raza, S. K., and Bieszczad, A.; "*Network Configuration with Plug and Play Components*"; Proceedings of the 6th IFIP/IEEE International Symposium on Integrated Network Management (IM'99), May 1999, Boston, USA; full text at URL: <http://www.sce.carleton.ca/netmanage/publications.html>.
- [8] Raza, S.K., "*A Plug-and-Play Approach with Distributed Computing Alternatives for Network Configuration Management*", Master of Engineering thesis, Carleton University, April 1999; Ottawa, Canada.
- [9] Raza S. K., Pagurek B., and White T.; "*Distributed Computing for Plug-and-Play Network Service Configuration*"; Submitted to IEEE/IFIP Network Operations and Management Symposium (NOMS'00); April 2000; Honolulu, Hawaii.
- [10] Raza, S.K., White, T., and Pagurek, B., "*Network Service Configuration with Jini: A Plug-and-Play Approach with Distributed Computing Alternatives*" Available at: <http://www.sce.carleton.ca/netmanage/publications.html>.
- [11] Sun Microsystems Inc.; "*Jini Architecture Specification*"; available at URL: <http://www.sun.com/jini/specs/>.
- [12] Tang, J., White, T., and Pagurek, B., "*Advanced Mobile Agent Architectures for H.323 IP Telephony*", To be presented at the workshop on Mobile Agents for Telecomm. Applications (MATA'99); Oct. 1999, Ottawa, Canada.
- [13] Vinoski, S., "*New Features of CORBA 3.0*"; Communications of the ACM, Vol. 41, No. 10, pp. 44-52, Oct. 1998.

TOWARDS A PLUG AND PLAY ARCHITECTURE FOR TELECOMMUNICATIONS

Finn Arve Aagesen, Bjarne E. Helvik, Vilas Wuwongse, Hein Meling,
Rolv Bræk and Ulrik Johansen

*Department of Telematics, The Norwegian University of Science and Technology
N-7491 Trondheim, Norway
finn.arve.aagesen@item.ntnu.no*

Abstract This paper presents an architecture specified within the project “Plug-and-play for Network and Teleservice Components” supported by The Norwegian Research Council. The hardware and software parts, as well as complete network elements that constitute a communication system, shall have the ability to configure themselves when installed into a network and then to provide services according to their own capabilities, the service repertoire and the operating policies of the system.

Plug-and-play components and functional objects are defined. A theatre analogy is chosen for the specification of the needed PaP support functionality. Plays define the functionality of the system. PaP components are realised by actors playing roles defined by manuscripts. An actor’s capabilities define his possibilities for playing various roles.

Keywords: Plug-and-play, Architecture, Telecommunication, Teleservices, Network Management, Smart Networks, Intelligent Networks, Active Networks.

1 INTRODUCTION

Plug-and-play (PaP) for telecommunications means that the hardware and software “parts”, as well as complete network elements that constitute a communication system have the ability to configure themselves when installed into a network (to plug) and then to provide services (to play) according to their own capabilities, the service repertoire and the operating policies of the system.

The concept PaP stems from the personal computing area. The objective of PaP in these systems is the handling of the plug-in of new devices and software into the desktop system. PaP simply means that you plug-in and then the system works. In these systems, the plugged in component as well as the framework has *predefined* functionality. We denote this kind of PaP as *static* PaP. As an example, static PaP in telecommunications occurs when a mobile plugs into the

network when switched on. The system provides static PaP with respect to the telephone service.

A more general kind of PaP is when the plugged-in unit has a set of basic capabilities, but its functionality is defined as a part of the plug-in procedure and it can be changed dynamically. We denote this kind of PaP as *dynamic* PaP. An example is a cellular phone which obtains the services it provides depending on its inherent capabilities, which user that logs on, and which network it is attached to.

With dynamic PaP, the definition of individual components and possibly, the overall structure of components can be changed on-line. One aspect of dynamic PaP is to change the services that a component provides. Another is to propagate the ability to use the service to all the service users.

The focus of this paper is on dynamic PaP. From now on the concept Plug-and-play means *dynamic* Plug-and-play. We foresee that dynamic PaP is needed to cope with the highly dynamic, heterogeneous and rapidly evolving networks and service provision of the future, and at the same time keep the networks manageable. The extra complexity introduced by PaP will make development more costly. However, it is expected that this cost is small compared to the benefits related to deployment, installation, operation, management, maintenance and evolution.

Section 3 defines the properties of a PaP system. Section 4 defines a PaP reference architecture which makes dynamic PaP possible. Section 5 discusses component categories and capabilities. Section 6 gives conclusions.

2 RELATED WORKS

The “grade of network intelligence” is here defined as *the efficient flexibility in the execution of teleservices and the efficient flexibility in the introduction of new teleservices*. The semantics of network intelligence is obviously related to time period in question. Intelligent Networks (IN) ([ITU92], [ITU93], [ITU97]), Telecommunication Information Networking Architecture (TINA) ([TINA94], [TINA95a], [TINA95b], [TINA95c], [TINA97a]), Mobile Agents and Active Networks ([Bies97], [Bies98a], [Bies98b], [Raza99], [Tenn96a], [Tenn96b], [Tenn97]) are all solutions aimed to improve the network intelligence.

The PaP system described in this paper is based on *mobile code*. Both Mobile Agents and Active Networks are based on mobile code. The mobile code can be based on *Code On Demand* (i.e. get the code when you need it), *Remote Evaluation* (i.e. send pieces of code to the destination, where the code can cooperate with other pieces of code), and *Mobile Agents* (i.e. the code has intelligence to take decisions of where to go to get executed). Our model will mainly be based on Code on Demand. Solutions to PaP within telecommunication networks has also been proposed based on Mobile Agents ([Bies97], [Raza99]). However,

the PaP functionality proposed are less comprehensive compared to the PaP functionality proposed in this paper. Note also that PaP as defined here has a wider scope of the network flexibility and adaptability than any of the above referred mobile agent and active network approaches.

3 PROPERTIES OF THE PAP SYSTEM

3.1 PaP components

The entities in the system subject to PaP are the PaP components. A *PaP component* is a real-world “concrete” reactive hardware or software module. PaP components can be:

- *Combined hardware/software* modules with one or more external hardware interfaces and a software platform capable of running PaP application software. Typical hardware modules are user nodes (i.e. user equipment with network connection), network nodes (i.e. circuit switches, packet switches, routers, cross-connects) and service nodes (i.e. service providing servers and databases).
- *Pure software modules* with interface to a software platform capable of running PaP application software. Typical software modules provide: Teleservice functionality, service and network management functionality, resource and QoS handling functionality, network protocol, routing and switching functionality and also plug-in and plug-out functionality.

Pure hardware modules are not feasible in the context of dynamic PaP. PaP components will exist together with components that do not have the PaP functionality. These are denoted as non-PaP components. In the following it is assumed that all PaP components interact via a common DPE (Distributed Processing Environment). PaP components can interact with non-PaP components. Figure 1 shows a general PaP component model.

A PaP component can *contain* other PaP components. All PaP components have a relationship to *PaP support*. The PaP support is based on database support. All PaP components can have a local data store not indicated in the figure. The logical relationships between the PaP components is partly related to how PaP is solved and partly to the specific functionality of the system.

As will be discussed in next section, the *PaP component* functionality is defined by a *functional object model* consisting of *functional PaP objects*. The relationship between two PaP components is the result of the external communication between the functional PaP objects that are used for the “realisation” of the PaP components. A functional PaP object is an instance of an PaP object type. More details on this subject will be given in Section 4.

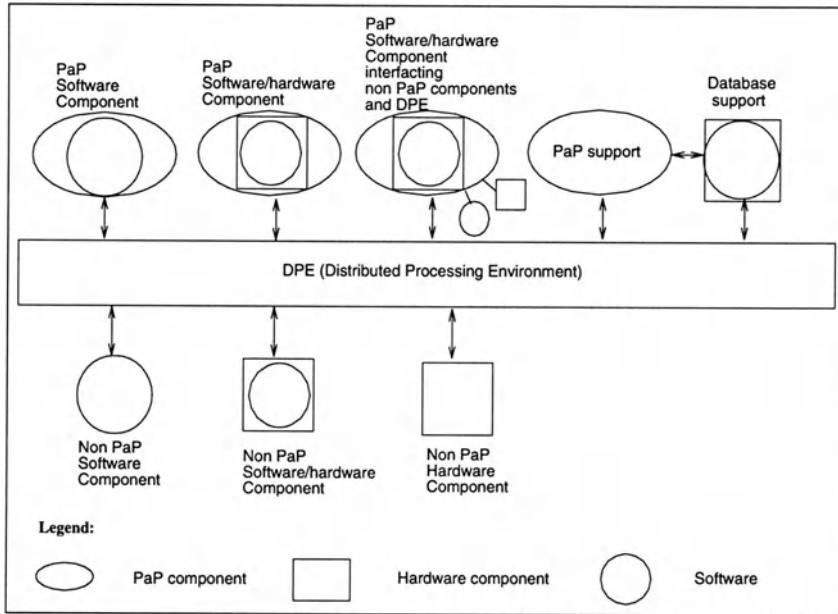


Figure 1 A general component model.

3.2 Requirements to the PaP system

PaP, as discussed here, is a technology intended for systems that shall be:

- flexible and adaptable,
- robust and survivable, and
- QoS aware and have resource control.

The goal of the plug-and-play technology is to significantly simplify and speed up the tasks of deployment, installation, management, evolution and maintenance. However, system structure and functionality to ensure a dependable and traffic handling capable solution are also important system properties. Note that neither the three "qualities" listed above, nor the requirements listed below are strictly disjoint. PaP properties that contribute to **flexible and adaptable** systems are:

1. A system structure and functionality that is not fixed, but which allows structural operations like adding, moving and removing components on demand, blocking/deblocking of components in response to failures, etc. Hence, it should be possible to add and change the functionality of the entire system by adding or changing (software) components.

2. *New components and their external capabilities* are found automatically. When a new component is plugged into a system, the system shall become aware of its presence. For instance, a new transmission line plugged between two routers should be recognised by the network and its capabilities, e.g. capacity, should be propagated to the routing algorithm of the network.

3. *Continuous adaptation to the environment and operation strategies/policies.* These are responses of a PaP system to changes in its operational environment and the conditions and rules it operates under. The system shall for instance be able to adapt to a change in the volume and interest of the traffic offered. Strategies/policies are set by the owner/operator of the system to guide its behaviour. Examples of strategies/policies are: priorities between services and/or users, and routing in the transport network.

4. *Recursive PaP functionality.* A PaP system may be a PaP component of a larger PaP system, i.e. aggregation of components. For instance, a router may be a PaP component of a local area network. This network may again be a component of a corporate network. With this recursiveness in mind, Property 3) also applies to the components of a system. Hence, they should have the ability to configure themselves when they are added into a system or network (ideally without human intervention) and have the ability to adapt themselves to changes in their surroundings in the system or particular events.

To be **robust and survivable** a PaP system must:

5. *Be based on a dependable distributed architecture* encompassing both the resources and the functionality of the system. The system shall not have a single point of failure, i.e. rely on unreplicated centralised resources and information. The system shall inhibit malicious and/or unauthorised modules to be plugged into the system.

6. *Reconfigure itself in the presence of failures* due to logical and physical faults. Failures due to physical faults will inevitably occur. The system shall be able to detect failed hardware components and automatically reconfigure itself so it can handle the workload and the functionality of the system to the extent that the physical resources in the system allow. Failures due to logical (software) faults are the more frequent. The system shall be able to detect these and to reinitialise failed components. It should also be able to prevent the propagation of errors in the system and import of errors from its environment.

7. *Provide continuous operation.* The system shall continue operation and shall not interrupt service provision more than strictly necessary under: plug-in

and plug-out of components, changes in services, operation and maintenance policies and manifestation of physical and logical faults.

To be **QoS aware and to provide resource control** a PaP system must:

8. *Negotiate QoS and allocate resources in an optimal way.* The basic types of resources of the system will be transfer (transmission) capacity, storage capacity and computation power. Limitations of these types of resource will restrict the QoS provided for a given workload and service mix. The system (and its components) shall negotiate with the users of its resources to ensure a good/optimal utilisation of its resources under the operating policies of the system.

9. *Provide monitoring of the resource utilisation* and dynamically take actions to improve it. Improvement of resource utilisation includes, reallocation of workload to other network elements and rearranging the location of pure software components. With respect to physical resources, it is expected that the system should be able to advice network operators and service providers about resources needed and changes required in the physical structure.

4 A PAP REFERENCE ARCHITECTURE

4.1 The functional object model

The PaP components are modelled and designed using object-oriented principles. PaP components are composed from (one or more) interacting instances of PaP functional objects, where each instance is defined by reference to an object type. ISO's reference model for Open Distributed Processing (ODP) [Duts96] defines the enterprise, computational, information, engineering and technical viewpoints. These viewpoints are tools for some kind of separation of the total system complexity. The computational and the engineering viewpoints are the viewpoint of *primary interest* with respect to PaP. The PaP components are basically engineering viewpoint objects. However, the PaP components have a computational viewpoint specification by the PaP functional objects, which are basically computational viewpoint objects.

We consider behaviour specification to be a computational aspect independent of the language used to express it. This because PaP always involves behaviour. The PaP of pure information will also be described as a part of the dynamic functionality. The computational model will also model the information which is subject to dynamic changes caused by the behaviour. In the PaP context, information models are supplementary models supporting the behaviour models.

A *functional PaP object* is a model of an aspect of a PaP component focusing on some behaviour aspect of the component. Most object-oriented systems

supports dynamic creation and removal of individual object instances. While this may be sufficient for static PaP, dynamic PaP requires in addition that:

- it is possible to change the definition of object instances and object instance structures, i.e. to change their type,
- to propagate the effect of such changes to involved object instances.

Thus, dynamic PaP require a PaP support system with the ability to manipulate type definitions, and to dynamically change object behaviours and object structures according to the changes of the corresponding types. This situation has many similarities with the theatre, which is chosen as a model to describe the support functionality of the PaP system. The basic structure of the PaP system is illustrated in Figure 2. This model also acts as a “bridge” between the PaP component specification and the functional PaP object specification. PaP components are “realised” by actors and actors are the entities “realising” the PaP functionality objects.

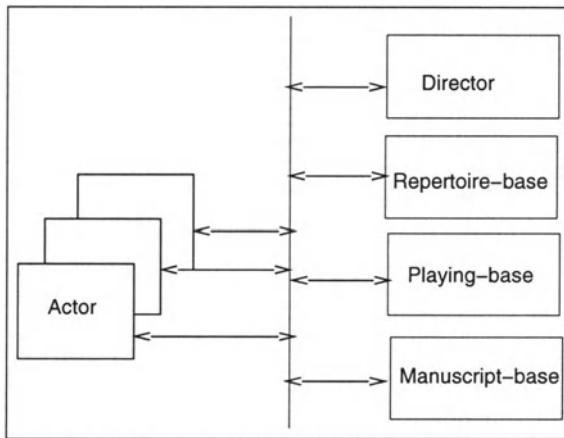
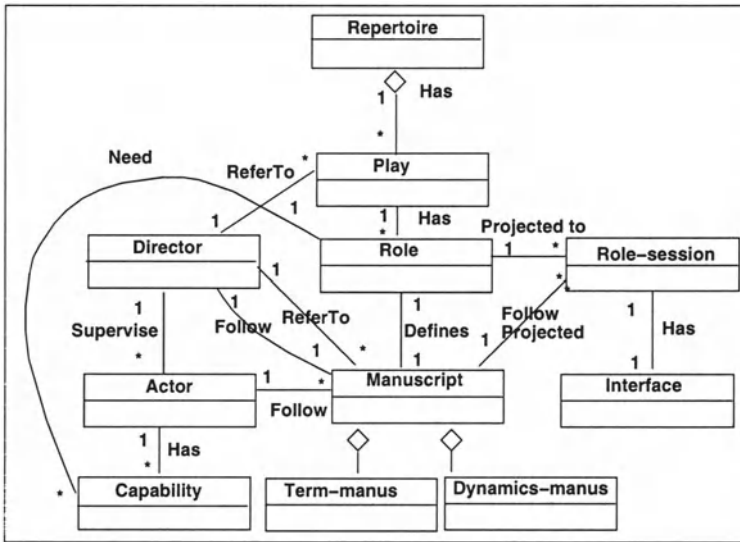


Figure 2 PaP system - Basic structure.

The model has many actor instances, one instance of a PaP-director, one instance of a repertoire-base, one instance of a manuscript-base and one instance of a playing-base. For simplicity, the system is in the present version modelled as a centralised system. An E-R model comprising important PaP concepts is illustrated in Figure 3.

An *actor* is a generic object with a generic behaviour. Actors are able to behave according to a *manuscript*. An actor also has a defined set of *capabilities*, which is the ability or power to do something. Capabilities are inherent properties of an actor, which can not be removed, replaced or copied without removing, replacing or copying the entire PaP actor. The capabilities are the result of the available hardware functionality connected to the hardware executing the



A *play* is a defined autonomous functionality. The play defines the context for relationships between PaP objects and their behaviour. One important PaP object functionality necessary to initialise any play is the *director*. A director behaviour is also defined by an instance of a play. An actor has three distinct behaviour phases: 1): the plug-in phase, 2): the play phase and 3): the plug-out phase. The director guides actors in the plug-in phase as well as in the plug-out phase. Important functionality related to the plug-in-phase is actor identification, actor access control, actor capability control and actor resource and QoS negotiation and allocation.

The *manuscript-base* has the manuscripts used by the actors to play their roles. The *playing-base* keeps a structural model of the instances of PaP objects that is actually playing. The *repertoire-base* keeps an overview of the potential plays and roles. Actors get an instance of a manuscript from the manuscript-base via the PaP-director. The manuscript of the PaP-director is also a part of

the manuscript-base. Behaviour and accordingly the manuscript is part of a *play*.

An actor is able to play various roles. A role is here similar to the theatre concept. The role is defined by a manuscript. Manuscript defines the total behaviour of an object. A *role session* is a projection of the behaviour of the actor with respect to one of its interacting actors. The role behaviour is specified by an instance of a manuscript. The manuscript specifies: cooperating PaP objects, how to reach the cooperating PaP objects, the interactions with the cooperating PaP objects and internal behaviour resulting from an incoming interaction.

There are two types of manuscripts: *term-manus* and *dynamics-manus*. A term-manus is a dictionary of terms which might be referred to by actors during their interactions to avoid ambiguity and misunderstanding. A dynamics-manus describes the overall dynamic characteristics of an actor. The dynamics-manus will also comprise the definitions of interfaces to be used during the play as well as needed capabilities. The dynamical behaviour is specified by an Extended Finite State Machine (EFSM).

Different from the theatre, and caused by the nature of telecommunication service providing systems, an actor can have its behaviour related to various plays at a time. However, an actor performs only one manuscript at a time. A PaP component, however, can handle various manuscripts by using various actors playing different manuscripts.

4.2 PaP support functionality

The following functionality is part of our dynamic PaP system: Play plug-in, Play changes plug-in, Dynamic detection of needs for actors/plays/roles, Actor plug-in, Actor behaviour plug-in, Actor play, Actor change behaviour, Actor behaviour plug-out, Actor plug-out and Play plug-out.

Play plug-in involves the updating of Repertoire-base and the Manuscript-base. The Play and role definition is assumed to have been done properly before the plug-in. The validation of possible conflicting interactions with other plays is a part of the new play plug-in problems to be considered. We denote this as play interaction problem. The service interaction problem [Najm99] discussed for years in the Intelligent networks community is of a similar nature.

Play-changes plug-in involves new object types and modified object types. Problems related to conflicting behaviour is as discussed for Play plug-in above. External changes must be considered as play plug-in. The challenge is to keep the system operational during the play-change installation period.

Dynamic detection of need of actors, plays and roles is specified in a manuscript. This is the "dynamic resource determination" solution. The advantage with this solution is that it is dynamically determined if, when, and how many actors and

plays are involved. Another possible solution can be to statically determine the needs for actors' prior to the playing of a certain play. In this case all need for, and allocation of actors and plays is done before the play start. The main advantage with this solution is to assure that all needed resources are available when starting a play. This is the "static resource determination" solution. A third solution, the "combined resource determination", may be to allow use of a combination of the two other solutions.

Actor plug-in is the creation of generic objects capable of playing various roles. An actor pending for play is initiated. An actor needs an actor-id. An actor is registered in the playing-base with a location-id and also its defined capabilities.

Actor behaviour plug-out is necessary procedures for taking an actor out of a play. This involves other actors and the playing-base. *Actor plug-out* is the local removal of actors pending for play and also the updating of the playing-base. *Play-plug-out* is the removal of the play from the repertoire-base and also the manuscript-base. The semantics of *Actor change behaviour* is dependent of the nature of the new behaviour. If the role is part of an existing "old" play, then actor change behaviour is equivalent to *Actor behaviour plug-in*. If the role is part of a brand new play, then actor behaviour plug-in must be preceded by play plug-in or play-changes plug-in.

The functions: *actor behaviour plug-in*, *actor play* and *actor behaviour plug-out* comprise the initialisation of a generic actor pending for a play, performing the real play, and finally making the actor pending for a new play. This functionality is denoted as the *basic PaP functionality* and is briefly described as follows. The actor is initialised by first activating its PaP-director. An actor negotiates with the PaP-director in order to obtain its behaviour. The negotiation is related to the limiting capabilities of the actor, the needed capabilities of the role and the optional choices of the role defined by the manuscript. Once the negotiation has been completed, the PaP-director will create an instance of a behaviour manuscript object with all necessary parameters bound particularly for the actor and then send it to the actor. The PaP-director also acts as a binding object which helps to establish communication or interactions among actors. After receiving an instance behaviour manuscript from the PaP-director, an actor will immediately start acting according to the specification described in the manuscript. From this point on in time the actor becomes autonomous and independent of the PaP-director.

5 COMPONENT CATEGORIES AND CAPABILITIES

The PaP component can be classified as clients and/or servers. A server can be one of more clients for other servers. This gives a *tree-structured* system of client-server relationships. A resource is a server with limited capacity.

Resources are important entities in a PaP system. A resource has a defined capability with respect to a service and quality of service (QoS). The PaP components can be categorised as

- *single objects*,
- *local objects aggregated* and
- *global objects aggregated*.

The single objects category is the simplest case where just one actor is plugged into a given play. An example is a new terminal server. For local objects aggregated, one component is plugged by using several actors at the same location. Examples are user nodes, network nodes and server nodes. In global objects aggregated, one component is plugged by using several actors at different locations. One example is a server requiring new software components at existing clients. In general a *teleservice* is *global* and corresponds to the result of behaviour performed by several distributed objects having various roles. Each teleservice is seen as a play. Each new play adds some new roles. Teleservices can not be defined separately and easily combined.

A capability has previously been introduced as the ability and/or power to perform functions or provide information. A capability of a PaP component is an:

- *inseparable*,
- *non-replaceable* and
- *non-replicable*

by the system itself functionality or attribute of the component. An attribute may be (private) information the component holds as well as performance characteristics. Hence, the functionality and parameters given an actor during the plug-in and during the play according to manuscripts are not capabilities. A component may have several capabilities. A certain set of (provided) capabilities is a prerequisite for filling a role as previously discussed.

The capabilities of an actor represents inherent, basic features that can be used to perform the roles assigned to it. The use of capabilities may be specified in a manuscript dynamically given to the actor, but the capabilities themselves can not be specified by such manuscripts. If an actor is seen as a machine that can be programmed using a manuscript language, then the semantics of this manuscript language will be related to the capabilities. Loosely speaking the capabilities provides the "instruction set" of the actor. They are not programmable (by dynamic PaP operations), but may be used as primitives in manuscripts.

The concept of capabilities are most easily understood for *combined hardware/software components*. In this case, the functionality of the hardware is embedded within the actor. The software part of the actor uses the functionality of the hardware to play roles in a system. As one example consider a hardware crypto module (e.g. a PCI board) and its accompanying software as a PaP component. The ability to perform fast and strong encryption (and decryption) based on the functionality of the hardware is capabilities of the corresponding actor(s). Actors from this component may take a variety of roles like encryption engine, decryption engine, digital signature generator and verifier and random number generator.

Pure software components also have capabilities, although these do not depend on a physical item. In these components the capabilities will be defined by:

- *Proprietary code*. This code is embedded within the actor and can not be (legally) separated from it or replicated in other ways. A software implementation of crypto functionality mentioned in the example above is an example of such code.
- *Private information*. This may be personal information or company confidential information which the actor may use but not reveal when it performs its functions. Such information will be found in actors with "agent functionality". For instance, an "agent-actor" playing in an auction type of service will have the bidding policies of its owner as one of its capabilities.

Needed capabilities are used together with play roles and related manuscript, while *provided capabilities* are used together with actors and components. During play performance the needed capabilities must be checked against the provided capabilities. This checking can be closely related to the interaction specifications for a play. Each interaction specification has defined needed capability requirements to the interacting actor. Only actors which can offer the needed capabilities are allowed, and able to perform the specified interaction. The conditions to be fulfilled before an interaction is possible due to requested capabilities, may be expressed as calculated optional sets of capabilities.

Since interaction between different plays are allowed, this also means that both play identifiers (play-id) and capabilities must be globally unique.

6 CONCLUSIONS

An architecture concept for dynamic Plug-and-play has been presented. The vision is a concept to be used to simplify and speed up the tasks of deployment, installation, operation, management, maintenance and evolution of various types of telecommunication equipment and services.

Plug-and-play components has been defined as real-world concrete reactive hardware and software modules. The PaP *component* functionality is defined by a *PaP functional object model*, consisting of *functional PaP objects*. A functional PaP object is an instance of an PaP object type. Dynamic PaP requires that it is possible to change the behaviour of an object and to propagate the effect of such changes. A functionality analogous to the theatre is chosen for the realisation of PaP. The most central issues are actors, roles, plays, manuscripts and capabilities. An actors capability defines his possibilities for playing various roles according to manuscripts.

The model presented is a step towards a complete architecture specification. The results presented are now applied for the specification and implementation of a PaP tele-school application demonstrator.

References

- [Bies97] Andrzej Bieszczad and Bernard Pagurek, Towards Plug- and Play Networks with Mobile Code, Proceedings of ICC'97, November 1997, <http://www.sce.carleton.ca/netmanage/publications.html>.
- [Bies98a] Andrzej Bieszczad and Bernard Pagurek and Tony White, Mobile Agents for Network Management, IEEE Communications Surveys, volume 1 number 1, 1998, <http://www.comsoc.org/pubs/surveys>.
- [Bies98b] Andrzej Bieszczad, S.K. Raza, Bernard Pagurek and Tony White, Agent-based Schemes for Plug-and-Play Network Components, Proceedings of the 3rd International Workshop on Agents in Telecommunications Applications, IATA'98, July 1998, <http://www.sce.carleton.ca/netmanage/publications.html>
- [Duts96] Joubine Dutszadeh and Elie Najm, Formal Support for ODP and Teleservices, Proceedings of the IFIP/ICC conference on Information Network and Data Communication, June 1996.
- [Najm99] Elie Najm, On Service Feature Interaction, Proceedings of Smartnet'99, Invited paper.
- [ITU92] ITU-T, Principles of intelligent network architecture, October 1992.
- [ITU93] ITU-T, Q1204: Intelligent network distributed functional plane architecture, March 1993.
- [ITU97] ITU-T, Intelligent network - Service plane architecture, September 1997.
- [Raza99] S. K. Raza and Andrzej Bieszczad, Network Configuration with Plug and Play Components, The Sixth IFIP/IEEE International Symposium on Integrated Network Management (to be presented) in 1999, <http://www.sce.carleton.ca/netmanage/publications.html>.
- [Tenn96a] David L. Tennenhaus, S.J. Garland, L. Shrira and M. Frans Kaashoek, From Internet to ActiveNet, Request for Comments, January 1996.
- [Tenn96b] David L. Tennenhouse and David J. Wetherall, Towards an Active Network Architecture, Computer Communication Review, Volume 26 number 2, April 1996.
- [Tenn97] David L. Tennenhouse, Jonathan M. Smith, David Sincoskie, David J. Wetherall and Gary J. Minden, A Survey of Active Network Research, IEEE Communications Magazine, Volume 35 no 1, 1997, pages 80-86.

- [TINA94] TINA Consortium, TINA-C Deliverable: Engineering Modelling Concepts, V2.0, December 1994.
- [TINA95a] TINA Consortium, TINA-C Deliverable: Overall Concepts and Principles of TINA V1.0, February 1995.
- [TINA95b] TINA Consortium, TINA-C Deliverable: Computational Modelling Concepts, V2.0, February 1995.
- [TINA95c] TINA Consortium, TINA-C Deliverable: Information Modelling Concepts, V2.0, December 1995.
- [TINA97a] TINA Consortium, TINA-C Deliverable: Service Architecture, V5.0, June 1997.

Distributed QoS Management for a Multiparty Video Conferencing System

Ines Beier, Hartmut Koenig

*Brandenburg University of Technology at Cottbus, Department of Computer Science,
PF 101344, 03013 Cottbus, Germany, Tel./Fax: +49-355-69-2236,
email: {beier,koenig}@informatik.tu-cottbus.de*

Key words: Multiparty interactive multimedia applications, multiparty video conferencing, group communication, distributed QoS management

Abstract: The QoS management for distributed interactive multimedia applications is currently mostly based on centralized approaches in which a single group server manages the QoS requirements for all participants. Group servers are simpler to handle, but they may become a weak point of the system due to server failures or performance bottlenecks. In order to avoid these problems and to better support the application in the endsystems a distributed QoS management at the application level seems to be more appropriate. In the paper we present such a distributed QoS management. It has been designed and implemented for the multiparty video conferencing system GCSVA. The basic idea of our approach consists in a decentralized calculation and assignment of the bandwidth quota for the speaker and listener whereby the receiving capacity of the most powerful participant is taken as measure. Less powerful receivers scale down the data rate of the incoming video streams by filtering according to their performance parameters. The consistency of the QoS data in the endsystems is ensured by means of a specialized group communication protocol which provides a reliable, atomic and ordered delivery service.

1. MOTIVATION

The increasing use of distributed interactive multimedia applications as joint editing or video conferencing for CSCW applications (*Computer Supported Cooperative Work*) requires appropriate mechanisms for the man-

agement and supervision of *Quality of Service* (QoS) requirements. Most existing video conferencing systems do not provide an extensive system support for the QoS and bandwidth management. In particular, they do not support the scaling of the video streams related to the number of conference participants and the performance of the used endsystems. The user is often forced to determine by „trial and error“ an acceptable service quality on a *best effort* provision.

In multiparty conference systems multimedia streams flow not only from one or a few selected sources to different receivers, but from each conference participant to all others, i.e. each participant¹ has to simultaneously process several arriving streams. In contrast to the compression of video streams, which is usually carried out in hardware, the decompression is implemented in software what considerably burdens the CPU². The simultaneous decompression of several video streams may lead to performance overload in the endsystems. Therefore, the QoS management has to take into account the performance capabilities of the endsystems. The negotiation of the QoS parameters in the traditional meaning [15] between all communication partners and the network for different streams becomes too complex. In addition, it has to be repeated when changing the composition of the conference or the network parameters. For that purpose, it seems to be more useful instead of applying a universal QoS framework to design special-purpose solutions for dedicated applications which exactly meet their requirements.

In this paper we present an approach for a distributed QoS management for a multiparty multimedia conference system. The approach has been designed and implemented for the video conferencing system *Gcsva* (*Group Communication and Scalability in Video conferencing over ATM*). *Gcsva* [3] has been designed for supporting CSCW applications and for teleteaching, especially for teleseminars. It is aimed at small closed discussions groups. *Gcsva* is characterized by distributed organization principles. It runs directly over ATM.

The remainder of the paper is structured as follows. In Section 2 we discuss the requirements on the QoS management for multiparty video conferencing systems and give an overview of related work. Especially, we argue the need for a distributed management. Section 3 discusses the floor control principles used in *Gcsva*. Section 4 deals with dynamic scaling of video streams and describes the solution applied. Section 5 presents the distributed QoS management approach of *Gcsva*. In Section 6 we describe how the QoS management has been implemented in *Gcsva*. We shortly describe the used group communication protocol GCP. Section 7 summarizes the results and outlines further research steps.

¹ In the following, we use the term participant to name a host participating in the conference.

² Hardware decompression can mostly handle only one stream.

2. REQUIREMENTS ON THE QOS MANAGEMENT FOR MULTIPARTY CONFERENCING SYSTEMS

Most existing conferencing systems scarcely support a flexible assignment of QoS parameters for different video and audio streams to control the bandwidth assignment. This is especially critical if the performance of the participants is very varying [13], [14]. It should be achieved that endsystems with varying performance parameters can handle multimedia streams with an acceptable service quality. They have to be protected against overload. Simultaneously, the waste of network resources should be avoided. Therefore, the video conferencing systems should provide means to dynamically scale down the multimedia streams according to the performance possibilities of the participant's hosts and the network.

In [9] it is proposed that the sources provide multimedia streams with 3 different qualities for the parameters frame rate, colour space and pixel resolution. The receivers can select the appropriate quality in accordance to their desires and performance parameters. Such an approach is useful for on demand services or teleteaching in which usually one source sends to several receivers. In a conferencing system in which each sender has to accept and send in realtime several streams of different quality this can overload the network and the participants.

A possible approach is the *receiver-driven layered multicast* described in [11]. Each source sends a stream consisting of different QoS levels. The receivers select the channels they want to accept and determine thus the quality of the received streams. In addition, the receivers can assign a weight to the senders [2]. These weights (priorities) are periodically announced. Thus more important sources (from the receiver point of view) can be distinguished from less important ones. In accordance to their weights the sources can transmit the different levels of their video streams (signal layers) over different network channels. For this approach though, a specific video compression method is applied.

In [6] a pure video conferencing system is presented which uses the so-called *group channel concept*. The group channel assigns a bandwidth quota to each participant whereby different quotas are defined for the speaker and the listeners. The total bandwidth for all video streams to be transmitted is chosen in a manner that the least performant host is not overloaded. Thus the receivers are protected against overload and network resources are saved. The shortage of this approach is that in case of very varying performance parameters of the participants the service quality is determined by the least powerful host. For powerful workstations, which are able to handle several videos with high quality, this is unsatisfactory. Here more flexible scalability approaches are required.

Another important design decision concerns the management scheme to be selected for the conferencing system. Most existing video conferencing systems use a centralized server for group and QoS management like the system in [6]. The server controls the whole conference. Usually, the participant which initiates the conference takes over the server role. When it leaves the conference mostly terminates. Centralized solutions are easily to implement because all management functions are concentrated in one system. This is especially true if sophisticated control algorithms are applied, e.g. for adapting the transferred video streams to the current performance situation (load) at the receiving endsystems. The main shortage of the centralized approach is a possible failure of the server which immediately terminates the conference. Furthermore, the server may become a performance bottleneck when many clients address to it simultaneously. Another problem is the passing over of the server function when the participant, which holds this function, leaves the conference, but a termination of the conference is not desired. To avoid these problems a distributed management approach seems to be more appropriate to us. A distributed approach is unavoidable when setting up video conferencing services over wide area networks as the internet. In [12] a distributed group management approach is also applied, but it does not include scaling mechanisms.

3. FLOOR CONTROL

In conferences there is usually only one speaker during a certain time slot. The other participants listen to its talk. The QoS management in Gcsva is based on this principle. It supposes that there exists always one defined speaker whose video and audio streams are handled with special care. This avoids large-scale methods for the speaker identification because the current speaker is constantly known. The assignment of the speaking right is regulated by means of a joint speaker queue. The head of the queue is the speaker. The other participants can shortly interrupt the speaker for questions or remarks. Their audio streams are merged with that of the speaker. This does not change, however, the speaker and any QoS parameters. The interrupt option can be switched off. There are other approaches to mimic the social behaviour in conferences as natural as possible [8]. In the current state of the development of Gcsva we do not consider any refined floor control mechanisms because we mainly focus on the elaboration of the distributed group and QoS management approach. This is planned for a later development phase.

The speaker queue is shown at each participant's screen. The queue can be manipulated by the participant. For example, it can remove itself from the

queue at any time when the discussion makes its remark unnecessary. The consistency of the queue at each participant's screen is ensured by the distributed group management (see below).

Since the speaker has a distinguished role in this conference scenario the video stream of the current speaker is transmitted with a higher quality of service than those of the other participants (listeners). Its video stream has a larger bandwidth quota than the video streams of the listeners. At receiver site it is only filtered if all listener's streams have scaled down (see Section 4.). Furthermore, the window size of the speaker is larger than that of the listeners (for which the same window size is used). The video streams of the listeners are handled with lower QoS on a best effort basis. These video streams are scaled down first at receiver site, if necessary up to a still image. The audio streams of all participants are transmitted with guaranteed QoS due to their low bandwidth requirement³.

4. DYNAMIC SCALABILITY OF VIDEO STREAMS

In a video conferencing system each participant receives video streams from all other participants. It has to decompress these streams before they are played out. The simultaneous decompression of several video streams can overload the endsystems. Scaling of video streams can reduce the decompression overhead. This can be done at distinct points whereby varying methods can be applied: the hierarchical, the sender-restricted and the receiver-restricted one. Hierarchical scaling is carried out in the network nodes according to the requirements of the receivers at the end of the path. Since our system is supposed to directly run over ATM this approach is not feasible.

In the sender-restricted method the sender adjusts the video stream in such a manner that all participants including the less powerful ones can accept the video stream. This method is applied for the group channel concept in [6]. The group channel defines one QoS level for the whole group limiting the total bandwidth of the conference to the performance of the slowest workstation of the group. The approach works well when the participants have nearly the same performance, but it considerably restricts the transmission quality on powerful machines when hosts with very different performance parameters are included in the conference.

The receiver-restricted method assumes that the network bandwidth is not the bottleneck. The hosts may send with the highest possible performance. The receivers scale them down depending on the number of incoming

³ In our prototype implementation we only could realize best effort QoS because the used FORE devices does not support guaranteed QoS (see Section 7).

streams and their current load. The approach, however, depends on the compression method applied. The scaling may also overload less powerful hosts.

In our approach we apply a combined scheme. The total bandwidth of the conference is related to that participant with the highest capacity. The bandwidth is split, as explained above, between all participants whereby the current speaker has a larger share. The participant's video streams are sent according to their quota on the total bandwidth. The video streams are adjusted by output filters if required. The receivers have to filter the streams depending on their performance parameters. The speakers stream is played out with the highest possible quality of service, i.e. with a higher frame rate and with a higher pixel resolution. It is less filtered than the listener's video streams. In overload situations only a still image of the listeners may be shown.

The number of video streams to decompress may change during the conference due to dynamic join and leave of participants. Therefore, the system has to dynamically adapt the filtering scheme. The total bandwidth and its splitting are renegotiated among the participants when a participant joins or leaves the conference.

Figure 1 gives an example of a specific scaling situation. We suppose that participant *c* has the highest capacity. It is able to process 60 frames per second (F/s). This is the total bandwidth. The speaker (participant *b*) gets a quota of 20 F/s, and the other participants (including participant *c*) may send with 10 F/s. Participant *a* has to filter from 60 F/s to 40 F/s. It depends on participant *a* how these 40 F/s are shared between the single video streams. For example, the speaker may be played out with 20 F/s and the other 3 participants get 6 F/s. Participant *d* shows only the speaker. For the other participants, it displays a still image.

The applied scaling method depends on the scaling purpose and the applied compression technique [7]. There are different possibilities such as temporal scaling, spatial scaling, frequency scaling, amplitude scaling, and colour space scaling. The objective of our scaling approach is to reduce the decompression effort and thus to decrease the calculation overhead at the receiver site of the video streams. Scaling methods which require large calculations are therefore not suitable. We use temporal and spatial scaling in the output filters (see Figure 3) for the scaling at sender site and only temporal scaling in the input filters at receiver site. For temporal scaling, frame dropping filters are employed. The compression method used is CellB. We first implemented filters for MPEG1 streams, but the achieved performance was too bad [5].

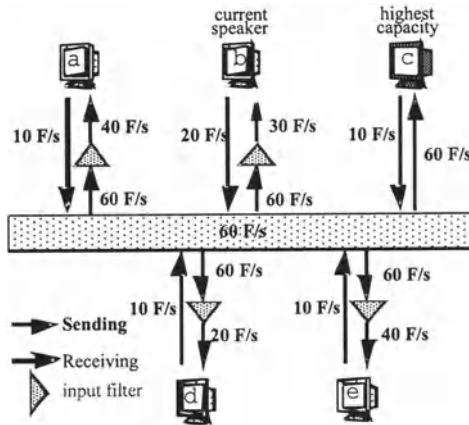


Figure 1. Example of a scaling situation regarding only the frame rate⁴

5. QOS MANAGEMENT

In this section we describe how the QoS parameters are calculated to support the above introduced scaling scheme. In order to make the decision process as simple as possible only the QoS parameters frame rate and pixel resolution are taken into account. The following decision is applied to the relation between speaker and listeners. The video stream of the speaker is recorded and sent with a higher frame rate and a higher pixel resolution. The other participants send with lower frame rate and pixel resolution whereby it is currently supposed that these parameters are equal for all listeners.

5.1 Calculation of the system parameters

A conference in Gcsva is successively set up, i.e. new participants join one after the other the conference. With each new joining all participants including the new one exchange their QoS parameters (frame rate and pixel resolution) by means of a *QoS-Request*-packet. The QoS parameters are provided by a monitor which observes the CPU load and derives the QoS values. It is supposed that no other application is running on the host, i.e. that the participant uses the computer only for the video conference.

Each participant indicates the quality in which it can receive the video stream of the speaker and the video streams of the other listeners. The number of the participants is known to the QoS management. The *QoS-Request*-packet of each participant contains the following values

⁴ The output filter are not shown.

$$\begin{aligned}
QoS\text{-}Request &= QoS_S, QoS_L \\
QoS_S &= (FR_S, FS_S) \\
QoS_L &= (FR_L, FS_L)
\end{aligned}$$

where FR_S and FS_S represent the desired frame rate and frame size (pixel resolution, pixel x pixel) for the video stream of the speaker. FR_L is the desired frame rate of the video streams of the listeners and FS_L the related pixel resolution.

After exchanging the *QoS-Request*-packets the QoS parameters for the whole system are calculated. They are related to the most powerful participant as explained in Section 4. Since frame rate and frame size can be different a measure must be found to compare the QoS requirements of the participants.

During a video conference only images of the head-shoulder perspective of the talking persons are usually transferred. Unlike movies, for instance, there are only a few changes in successive frames. The influence of inter-frame coding on the transmission rate can be therefore neglected in the following discussion. The transmission rate of the video streams is thus determined by the frame rate and the pixel resolution. The dependencies are linear (see Figure 2).

In order to express that the speaker possesses the highest priority we introduce weights. The speaker is assigned a double value compared to the listeners. All listeners possess the same weight. The sum of the weights of all participants is 1. Let n the number of participants and W_x the weight of participant x then

$$W_S = \frac{2}{n+1} \quad \text{and} \quad W_L = \frac{1}{n+1}$$

are the weights for the speaker and the listener.

In the following we determine the value C which is taken as measure to compare the QoS requirements of the participants. The QoS requirement of participant i is

$$C_i = (W_S \cdot FR_{S,i} \cdot FS_{S,i}) + (n-1) \cdot (W_L \cdot FR_{L,i} \cdot FS_{L,i})$$

If the local video stream is not shown then $(n-2)$ has to be introduced instead of $(n-1)$.

The system QoS configuration QoS^* related to sending is determined by the QoS requirement C_k of participant k for which the maximum C_{max} of all C_i is calculated:

$$QoS^* = (QoS_S^*, QoS_L^*) = ((QoS_S, QoS_L)_k \mid (C_{max} = C_k = \max(C_i, i = 1(1)n, (1 \leq k \leq n)))$$

QoS^* has to be recalculated if a new participant joins the conference or a participant leaves it.

After changing the speaker the new speaker has to send with the new QoS parameters QoS_S^* , all other participants including the preceding speaker

with the listener's parameters QoS_L^* . All participants have to take this change into account when receiving the video streams.

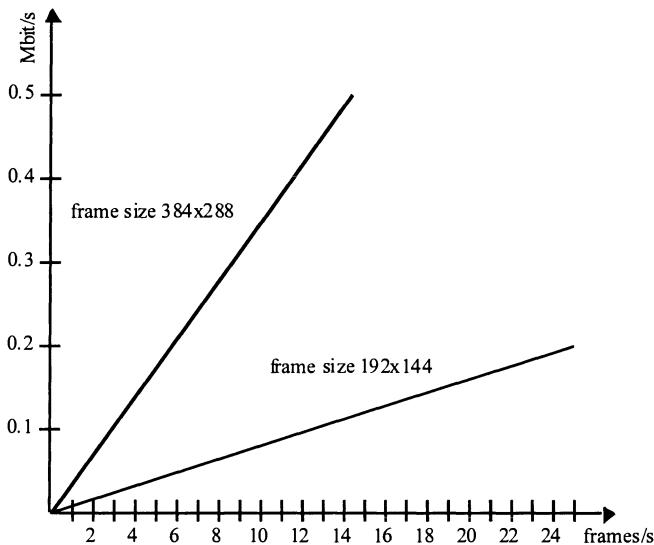


Figure 2. Influence of frame rate and frame size on the bandwidth⁵

5.2 Calculation of the local QoS parameters

After receiving the valid system parameters $QoS^* = (QoS_S^*, QoS_L^*) = ((FR_S^*, FS_S^*), (FR_L^*, FS_L^*))$ for the sending of the video streams each participant has to determine for itself how it must filter the incoming video streams. The frame size of the arriving (already recorded) video stream can be only changed by a partial decompression. This requires an additional computing effort which should be avoided at receiver site. Therefore, we decided that all participants receive and handle the incoming video streams with the system frame sizes FS_S^* and FS_L^* , respectively. Thus scaling reduces to adjusting the frame rates in the input filters. Let $FR_{S,i}^+$ the frame rate of the video stream of the speaker which can be accepted by participant i then the reduction is calculated based on the linear dependency between frame rate and frame size (see Figure 1) by the following formula:

$$FR_{S,i}^+ = \frac{FS_{S,i}}{FS_S^*} \cdot FR_{S,i}$$

⁵ The measurements are carried out on a SparcStation 20 with the compression method CellB.

The filter has to scale down the frame rate of the incoming speaker's stream from FR_S^* to $FR_{S,i}^+$. The same scaling principle is applied to the frame rate of the video streams of the listeners $FR_{L,i}^+$.

These parameters have to be recalculated for each change of the composition of the group or of the speaker, respectively.

6. IMPLEMENTATION OF THE APPROACH

6.1 Architecture of the GCSVA system

The implementation of the presented approach requires a management component at each participant's host as well as a protocol support for their interaction. Figure 3 shows the architecture of the Gcsva at the participant's host. Because video and audio streams require a service which is very different from that needed for transmitting group communication data we divided the architecture in two parts: a data handling part and a signalling part. They are supported by two different protocols: the **MCM-TP** (*Multicast Continuous Media Transport Protocol*) for the transmission of the audio and video streams, and the **GCP** (*Group Communication Protocol*) for the group communication. Both protocols run directly over ATM Adaptation Layer 5 (AAL5) [10].

The data handling part comprises all tasks for handling the incoming and outgoing video and audio streams. This is done in the **Video** and **Audio Manager**. The video manager contains also the **Output Filter** to adjust the outgoing video streams according to the system configuration QoS^* . The transmission of the audio and video data from one sender to the other participants is carried out by means of MCM-TP. It provides a connection oriented unreliable multicast service. The MCM-TP layer contains the **Input Filter** which scales down the incoming stream of the speaker from FR_S^* to $FR_{S,i}^+$ and those of the listeners from FR_L^* to $FR_{L,i}^+$, respectively.

The signalling part is responsible for the group and QoS management. It consists of the QoS Manager, the Monitor, and the Group Management Module. The **QoS Manager** calculates the QoS parameters for the system configuration QoS^* , and the local filtering parameters $FR_{S,i}^+$ and $FR_{L,i}^+$ as described above. The work of the QoS manager is supported by a monitor. The **Monitor** observes the current CPU load. It periodically sends this information to the QoS manager for calculating the filter parameters. The recalculation of the QoS parameters is triggered by the Group Management Module when the speaker changes, or a participant joins or leaves the conference. The QoS management at ATM level is organized as follows. The QoS parameters for the ATM connections are determined during connection

set up. They cannot be changed afterwards. A reservation of network resources for the transmission of the streams among all participants in best quality would waste network resources and cause higher cost. Since in Gcsva all participants do not always send with the highest possible quality only an average quality is required for the QoS parameters during connection set up at ATM level.

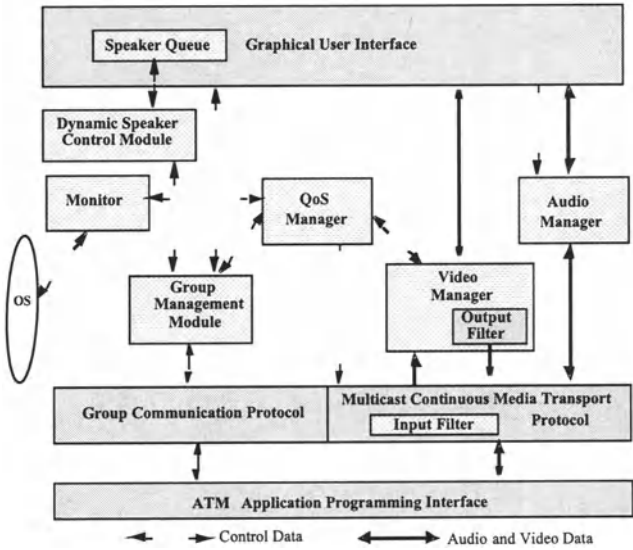


Figure 3. Gcsva architecture at each participant's host

The **Group Management Modules** supervise the state of the group. They exchange messages on the composition of the group (join and leave of participants) as well as messages for controlling the group (entering the speaker queue, removing from the speaker queue, exchange of QoS requirements). The Group Management Modules are supposed to free users from any QoS management because users do not usually possess a deep knowledge of QoS parameters and of the relation between them.

For the communication among the Group Management Modules, the multicast protocol GCP was designed. It ensures the consistency of the QoS parameters and the group management information at participants' site. The GCP is the basis of our decentralized group management and described separately in the next subsection.

6.2 **The Group Communication Protocol (GCP)**

The GCP protocol was designed after carefully analysing and in part adapting ideas from existing multicast protocols and group communication approaches. In order to ensure the consistency of the group management data

in the Group Management Modules the supporting protocol has to fulfil the following requirements:

- *reliability*
In contrast to the transmission of video data where some frames may be lost, distorted or discarded, the exchange of control data has to be reliable. Messages may not be distorted, lost, or out of order.
- *atomicity*
A message must be either delivered to all participants or to none.
- *ordered delivery*
If the order affects the results, messages of different senders have to be delivered to different receivers in the same order. Depending on the application, different levels of ordering may be required.
- *dynamic join and leave*
Participants should be allowed to join and to leave the conference at any time.

The GCP meets this requirements. It provides a reliable, atomic, and ordered delivery service.

To provide an ordered delivery GCP applies a token-based mechanism similar to [1]. All participants form a logical ring on which the token rotates. Only the tokenholder is allowed to send. All participants have to acknowledge the reception of these PDUs. Unacknowledged PDUs are retransmitted after timeout up to three times. The token is forwarded when all acknowledgements have been received. The reception of the token has to be acknowledged as well. If there are no messages to send the token is forwarded immediately.

If there are any outstanding acknowledgements after a certain period of time, the so-called *forced leave* mechanism is triggered. It removes these participants from the group (i.e. they have to leave the conference). Thus it is guaranteed that atomicity is achieved because the remaining participants received the PDU. Token loss and duplication are handled in a similar way.

Since only one messages exchange (data PDU delivery and related acknowledgements) takes place between sender and receivers, crossings and overtakings of messages between two token movements are impossible. All participants receive all messages in the same order. This ensures the totally ordered delivery.

The rotating token mechanism further guarantees fairness between all participants. The shifting of the token even if no data has to be transmitted supports the early detection of failures of the participants.

A more detailed description of the design and of the features of GCP is given in [4].

7. SUMMARY AND OUTLOOK

In the paper we have presented a QoS management concept for distributedly organized video conferencing systems which supports a dynamic scalability of the video streams. It was designed and implemented for the multiparty conference system Gcsva. The distributed organization of a video conferencing system avoids the issues of centralized solutions such as server failure, performance bottlenecks and function handover. A distributed approach requires a more ambitious design which causes an organizational overhead. Future multiparty conferences over wide area networks, especially the internet, will require a distributed management. We have presented an approach how the QoS parameters for the filtering of the video streams can be calculated and have shown how this is implemented in Gcsva. The kernel of the approach is the GCP which ensures the consistency of the distributed management data.

The QoS management forms the basis for a dynamic scaling of the video streams depending on the number of participants, their performance, and their current load. The advantage of the proposed solution consists in the automatic adjustment of the QoS requirements without any interference of the users. By combining sender- and receiver-restricted scaling it is possible to adapt the QoS of the video streams to the requirements of the individual receivers. In a next research step a broader range of QoS parameters will be considered to make the QoS management more flexible. We will also investigate possibilities to increase the users influence on the QoS management.

A prototype implementation of Gcsva on four SUN workstations with different performance characteristics has been finished. The experimental ATM network consists of a switch ASX200WG (FORE systems) and ATM interface cards SBA200. Currently measurements for evaluating the performance behaviour of the system are prepared.

In a joint project between the Hong Kong University of Science and Technology (HKUST) and our university we are currently developing the video conferencing system Octopus [16]. The Octopus approach aims at connecting local video conferences to a global video conferencing via wide area networks, e.g. the internet. It supports a dynamic adaption to workload changes to maintain smooth videos. The interactions between the local video conferencing systems are controlled by group coordinators which have to be introduced as interface in the local conferences. The group coordinators use for communication the GCP which is implemented over IP for this purpose.

REFERENCES

1. Agarwal,D.A.: Totem: A Reliable Ordered Delivery Protocol for Interconnected Local-Area Networks, Ph.D Thesis, University of Santa Barbara,1994
2. Amir,E.; McCanne,St.; Katz,R.: Receiver-driven Bandwidth Adaptation for Light-weight Sessions, ACM Multimedia, 1997, Seattle, WA.
3. Beier,I.; König,H.: Gcsva - A Multiparty Video conferencing System with Distributed Group and QoS Management, 7th International IEEE Conference on Computer Communications and Networks (IC3N'98), Louisiana, 1998, 594-598
4. Beier,I.; König,H.: A Protocol Supporting Distributed Group and QoS Management, Proc. IEEE Conference on Protocols for Multimedia Systems - Multimedia Networking PROMS-MmNet'97, Santiago, Chile, 1997, 213-222
5. Beier, I.; Fiedler, J.; König, H.: A Multimedia Conferencing System with Distributed Group and QoS Management. 4th Int. Symposium on Communications ISCOM'97, Taiwan, 1997, 134-138
6. Chan,D.L.S.; Chanson,S.T.: Scalability Support for Multiparty Multimedia Communications, in ACM/Springer Multimedia Systems Journal, 6(1998)2, 75-87
7. Delgrossi,L.; Halstrick,C.; Hehmann,D.; Herrtwich,R.G.; Krone,O.; Sandvoss,J.; Vogt,C.: Media scaling in a multimedia communication system, Multimedia Systems, 2(1994), 172-180
8. Dommel,H.-P.; Garcia-Luna-Aceves,J.J.: Floor control for multimedia conferencing and collaboration, Multimedia Systems (1997)5, 23-38
9. Fischer,S; Hafid,A.; v.Bochmann,G.; de Meer,H.: Cooperative QoS Management for Multimedia Applications, 4th IEEE Int. Conf. on Multimedia Computing and Systems (ICMCS'97), Ottawa, 1997, 303-310
10. ITU-T: ITU-T Recommendation I.362, B-ISDN ATM Adaptation Layer (AAL) functional description, Geneva, 1993
11. McCanne,St.; Vetterli,M.; Jacobson,V.: Low-complexity Video Coding for Receiver-driven Layered Multicast, IEEE Journal on Selected Areas in Communications 16(1997) 6, 983-1001
12. Schubert,I.; Sisalem,D.; Schulzrinne,H.: A Floor Control Application for Light-Weight Multicast Conferences, Int. Conference on Telecommunications ICT'98, 1998, Greece
13. Sun Microsystems: The Complete Guide to ShowMe 2.0.1, 1994
14. Turetti, T.; Huitema, C.: Video conferencing on the Internet, ACM Transactions on Networking, 4(1996)3
15. Vogel, A.; Kerheve,B.: QoS Negotiation for Distributed Multimedia Presentational Applications, Workshop on Distributed Multimedia Applications and QoS-Verification, Montreal, Quebec, 1994
16. Chanson,S.T.; Hui,A.; Siu,E.; Beier,I.; Koenig,H.; Zuehlke,M.: Octopus - A Scalable Multiparty Global Video conferencing System. Accepted for 8th International IEEE Conference on Computer Communications and Networks (IC3N'99), Boston, Oct. 1999

PERFORMANCE EVALUATION OF AN IP VOICE TERMINAL

Harri Marjamäki & Raimo Kantola

Harri Marjamäki, M.Sc.

Raimo Kantola, Ph.D., Professor of Telecommunications Technology

Key words: Voice over IP, Playout algorithms, Performance evaluation

Abstract: This paper examines the issues related to the transmission of voice over packet networks using the Internet Protocol (IP). We focus on studying the delays that are generated in the terminal, which is a Unix workstation equipped with IP voice application software.

Delay components in the terminal are presented. We measure the processing delays in the terminal using different audio codecs and measure the end-to-end delays using different scheduling parameters for the IP voice application. A significant part of the delay is shown to be caused by buffering at the receiving terminal. This is a feature of the application software and can be removed by modifying the source code.

We also make a comparison of adaptive algorithms that are used to calculate the playout times of the voice packets. Algorithms are simulated using different network loads and thus different delay distributions of the voice packets in an Ethernet. We present a new playout algorithm, which is a combination of two existing algorithms. This algorithm is shown to outperform the other two real-time algorithms that are compared in our studies.

1. INTRODUCTION

The Internet provides a simple single class best effort service. From a connection's point of view, the best effort service amounts in practice to offering a channel with time-varying characteristics such as delay and loss distributions. These characteristics are not known in advance since they depend on the behaviour of other connections throughout the network. A variety of audio tools have been available for a few years, and they have been used to audiocast conferences. Experimental evidence suggests that, although the quality of the audio delivered by Internet tools has improved, audio quality is still mediocre in many audio conferences. This is clearly a concern since audio quality has been found to be more important than video quality or audio/video synchronization to successfully carry out collaborative work.

For audio quality in packet audio applications, the main concerns are the delay and delay variance. In earlier studies [1], [2] it was noted that in LANs and campus networks where network caused delay and delay variance was relatively small, most of the end-to-end delay was accumulated in the terminals.

In the terminal, delay is accumulated both by the hardware and the software. In the audio hardware, voice samples are A/D converted at the sender and D/A converted at the receiver. In the packet audio application software processing delay is introduced. Processing delay is very much dependent on the used speech codec. Some codecs, like PCM codec have very little to do and introduce very little delay whereas, for example, a GSM codec requires excessive computation and causes significantly more processing delay. Buffering of voice samples is necessary both at sending and receiving end. Buffering delays are introduced both in the audio hardware and in the packet audio software. Delays are introduced also by the operating system because it has to assign processor time also to other processes that are simultaneously running in the terminal.

The network caused delay variance has to be smoothed in the application software in order to preserve the sound quality. Voice packets are buffered at the receiver and they are played out periodically. The algorithms used to calculate the appropriate playout time for each packet of voice are called playout algorithms.

The rest of the paper is organized as follows: Chapter two explains playout delay adaptation: the mathematics and algorithms used in delay estimation. Chapter three reports performed measurements and obtained results. Chapter four presents conclusions and topics for future work.

2. PLAYOUT DELAY ADJUSTMENT

Packet audio tools operate by periodically gathering audio samples generated at the sending host, packetizing them, and transmitting the resulting packet to receiving site(s). For efficiency, the source audio is typically divided into talkspurts (periods of audio activity) and silence periods (periods of audio inactivity, during which no audio packets are generated). In order to faithfully reconstruct the audio at the receiving site, data in packets within a talkspurt must be played out in the same periodic manner in which packets were generated.

If the underlying network is free of variations (jitter) in packet delays, a receiving site can simply play out an audio packet as soon as it is received. However, jitter-free, in-order, on-time packet delivery rarely, if ever, occurs in today's packet-switched networks. In order to compensate for these variable delays, a smoothing buffer is thus typically used at a receiver. Received packets are first queued into the smoothing buffer and the periodic playout of packets within a talkspurt is delayed for some amount of time beyond the reception of the first packet in the talkspurt. We refer to this delay as the playout delay of the talkspurt. Clearly, the longer the playout delay, the more likely it is that a packet will have arrived before its scheduled playout time. Excessively long playout delays, however, can significantly impair human conversations. There is thus a critical tradeoff between the length of playout delay and the amount of loss (due to late packet arrival) that is incurred. Generally, delays between talkspurt generation and receiver playout of less than 400 ms [3] and a loss percentage of up to 5% [4] are considered to be quite tolerable in human conversations.

The talkspurt playout delays themselves can be either fixed for the duration of the audio session, or adaptively adjusted. In the Internet, end-to-end delays fluctuate significantly and a constant, non-adaptive, playout delay would thus likely yield unsatisfactory audio quality for interactive audio applications. There are two approaches for adaptive playout adjustment: per-talkspurt and per-packet adjustment. The former uses the same playout delay throughout a talkspurt (and, as a result, faithfully reconstructs the original periodic nature of the received audio data from the sender), but allows different playout delays from one talkspurt to another. While this may result in artificially elongated or compressed silence periods, this is not noticeable in played out speech if the change is reasonably small [5]. In the latter approach, the playout delay varies from packet to packet. A per-packet adaptive adjustment introduces gaps inside talkspurts and is cited as damaging to the audio quality [6].

2.1 End-to-end delay characteristics

Previous studies [7] have indicated the presence of “spikes” in end-to-end Internet delays. A spike constitutes a sudden, large increase in the end-to-end network delay, followed by a series of packets arriving almost simultaneously, leading to the completion of the spike.

With periodically generated packets, the initial steep rise in the delay spike and the linear, monotonic decrease after the initial rise, is due to “probe compression” – the accumulation of a number of packets from the connection under consideration (the audio session, in our case) in a router queue behind a large number of packets from other sources. Probe compression is a plausible conjecture about the cause(s) of delay spikes.

2.2 Performance of a playout algorithm

The tradeoff between the average playout delay and loss due to late packet arrival is used as the performance measure in comparing one adaptive playout delay adjustment algorithm with another. Loss and delay are considered on a per-packet rather than per-talkspurt basis for two reasons. First we note that the lengths of talkspurts depend on silence detection algorithms and their parameters. Per-talkspurt results are thus closely tied to the silence detection algorithm used. More importantly, different talkspurts have different lengths.

Here the end-to-end application-to-application delay is defined to be the difference between the playout time at the receiver and the generation time at the sender. We refer to Figure 1 to show the timing information of audio packets and formally define the average playout delay [9].

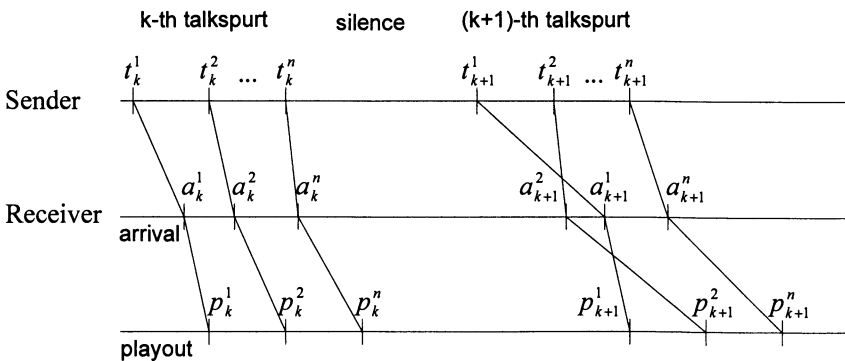


Figure 1. Timings associated with the i -th packet in the k -th talkspurt

Consider a trace consisting of M talkspurts. We define the following quantities:

- t'_k : sender timestamp of the i -th packet in the k -th talkspurt
- a'_k : receiver timestamp of the i -th packet in the k -th talkspurt
- n_k : number of packets in the k -th talkspurt. Here we only consider those packets actually received at the receiver.
- N : total number of packets in a trace,

$$N = \sum_{k=1}^M n_k \quad (1)$$

The playout time of a packet depends on which algorithm is used at the receiver to estimate the playout delay of the packet. Consider a playout algorithm A . Then $p'_k(A)$ is the playout timestamp of the i -th packet in the k -th talkspurt under A . If the i -th packet of the k -th talkspurt arrives later than $p'_k(A)$ (i.e., $p'_k(A) < a'_k$), it is considered lost. Otherwise, it is played out with the playout delay of $(p'_k(A) - t'_k)$. Let $r'_k(A)$ be an indicator variable for whether the i -th packet of the k -th talkspurt arrives before its playout time, as computed by playout algorithm A :

$$r'_k(A) = \begin{cases} 0, & p'_k(A) < a'_k \\ 1, & \text{otherwise} \end{cases} \quad (2)$$

The total number of packets played under algorithm A is denoted as $N(A)$ and computed using $r'_k(A)$:

$$N(A) = \sum_{k=1}^M \sum_{i=1}^{n_k} r'_k(A) \quad (3)$$

Then the average playout delay of those played-out packets is defined as:

$$\frac{1}{N(A)} \sum_{k=1}^M \sum_{i=1}^{n_k} r'_k(A) (p'_k(A) - t'_k) \quad (4)$$

If there are N packets in a trace and, among them, $N(A)$ packets are played out under algorithm A , the loss percentage l is:

$$l = \frac{N - N(A)}{N} * 100 \quad (5)$$

2.3 Some playout algorithms

In this section we present 4 different playout algorithms. Algorithms 1 and 2 are originally reported in [8] and algorithm 3 is suggested in [9]. Algorithm 4 is a combination of algorithms 1 and 3. The mathematical descriptions of the algorithms are not presented because of the lack of space.

2.3.1 Algorithm 1

This algorithm is based on stochastic gradient algorithms used in estimation and control theory [5], and operates by estimating two statistics characterizing the network delay incurred by audio packets: the delay itself, and a variational measure of the observed delays. Each of these estimates is recomputed each time a new packet arrives.

Algorithm 1 is a linear filter that is slow in catching up with a change in delays, but is good at maintaining a steady value, when the gain of the estimator $(1-\alpha)$, is set to be very low.

2.3.2 Algorithm 2

Algorithm 2 has two modes of operation, depending on whether a spike has been detected. In normal mode, it operates like algorithm 1 with a different gain, but in spike-detection mode, the playout delay is updated differently.

Algorithm 2 works as follows: It is checked if the delay between consecutive packets at the receiver is large enough for it to be called a spike. On detection of a spike, we enter the spike mode and “follow” the spike. Thus, in the spike mode, we allow our estimate to be dictated only by the most recently observed delay values.

2.3.3 Algorithm 3

The key idea behind this algorithm is to collect statistics on packets that have already arrived and to use them to estimate the playout delay. Instead of using the linear filter mechanism, each packet's delay is logged and the distribution of packet delays is updated at every packet arrival. When a new talkspurt starts, the algorithm calculates a given percentile point q in the distribution function of the packet delays for the last w packets, and uses it as the playout delay for the new talkspurt. As in algorithm 2, it detects spikes and behaves accordingly: once a spike is detected, it stops collecting packet delays and follows the spike until it detects the end of a spike. Upon detecting the end of a delay spike, it resumes its normal operation.

2.3.4 Algorithm 4

With algorithm 3, the problem is that it first needs to collect some delay statistics. In a real-time implementation it is not possible to use algorithm 3 from the beginning of a call. Therefore we can combine together algorithms 1 and 3 and the resulting algorithm is called here algorithm 4. The idea is

simply to calculate the playout delays with algorithm 1 until we have received 5000 voice packets and then switch to algorithm 3. The switch can't be done earlier because then algorithm 3 would probably give worse results than algorithm 1. The extension to algorithm 1 is that we use the two operating modes from algorithm 3 since the beginning of the call.

3. MEASUREMENTS AND RESULTS

In this chapter we present the results of our experiments. The presentation is divided in three parts. First part handles the processor time consumption in an IP voice terminal, the second part handles the end-to-end delays in an IP voice connection, and in the third part different playout algorithms are compared. The used VoIP client in all experiments is Nevot (Network Voice Terminal) [10]. Nevot was chosen because of its high configurability compared with other VoIP tools. Nevot provides for example a feature to switch off the additional playout delay for testing purposes. Nevot's source codes are freely downloadable in the Internet, which made it possible to compile it from sources and examine the program implementation in practise. Thirdly, Nevot provides a debugging option to record sent and received RTP-headers which is useful for playout algorithm simulations.

3.1 Measurement of the CPU consumption

We wanted to know how much processor time is consumed by Nevot using different audio codecs. In order to do this, version 3.35 of Nevot was compiled from sources using -p flag. This enables us after running the program by using the prof command to produce a profile file which shows for each external text symbol the number of times that function was called and the average amount of time per call.

Nevot provides the following speech codecs: PCM 64 kbit/s, ADPCM 32 kbit/s, GSM 13 kbit/s and LPC 4,8 kbit/s. PCM and ADPCM are computationally light codecs compared to GSM and LPC, which are computationally rather intensive.

3.1.1 Setup of the measurement

We had two Sun Ultra Enterprise 1 workstations connected to a 10BaseT Ethernet. Both workstations were running Nevot version 3.35 for 5 minutes in each measurement. Packet sizes were set to 20 ms and silence detection

was disabled so that both clients were continuously sending and receiving packets.

3.1.2 Results of the measurements

Consumption of CPU time for different codecs is presented in table 1. These results show that the processing delay increases with the complexity of the codec. The values in Table 1 include both coding and decoding in one workstation.

Table 1. Used CPU time in milliseconds and percent of used CPU time from total CPU time with different audio codings

<i>Used CPU time</i>	<i>PCM</i>	<i>ADPCM</i>	<i>GSM</i>	<i>LPC</i>
<i>in milliseconds</i>	0.26	0.49	1.14	1.83
<i>in percent</i>	1.3	2.5	5.7	9.1

3.2 Measurement of the end-to-end delay

In this measurement we had two Sun Ultra Enterprise 1 workstations with SunOS 5.5.1 connected to a 10BaseT Ethernet. Both workstations were running Nevot version 3.35. Delays were measured with four different audio codings. For each audio coding both half-duplex and full-duplex traffic was measured. We also used two different process priorities for Nevot to see if the operating system has some contribution to the delay. Used priorities were normal time-shared class priority with user priority 0 and realtime priority with the highest possible priority 59 and time slice of 1 second. Each type of measurement was repeated 10 times. Packet size was set to 20 ms in all measurements.

3.2.1 Results of the measurements

Delays were measured 10 times for each configuration. Table 2 presents the averaged results.

Table 2. Average end-to-end delays in different audio codings

	<i>TS, half duplex</i> <i>[ms]</i>	<i>TS, full duplex</i> <i>[ms]</i>	<i>RT, half duplex</i> <i>[ms]</i>	<i>RT, full duplex</i> <i>[ms]</i>
<i>PCM</i>	32.7	32.7	31.3	31.4
<i>ADPCM</i>	33.4	34.9	31.6	32.0
<i>GSM</i>	34.7	33.5	32.5	33.6
<i>LPC</i>	35.0	36.4	33.0	35.2

3.2.2 Analysis of the measurements

Measurements were done over a non-loaded LAN and network caused delay was measured with Ping and was shown to be constantly around 0.5 ms. The framing delay of 20 ms is included in all results.

Hardware caused delays were measured by directing the signal from microphone input to headphones output by Audio tool program. Thus, a 1 kHz square-wave input signal was A/D converted and then D/A converted in the audio hardware. Input signal was connected to the first channel of an oscilloscope and output signal to the second channel of the oscilloscope. HW delay was measured several times and was constantly 2.2 ms.

If we take for example full duplex measurement with real time priorities and PCM audio coding and subtract 31.4 ms - 20 ms (framing delay) - 2*0.26 ms (processing delay) – 0.5 ms (network delay)- 2.2 (HW delay) we are left with 8.2 milliseconds. This delay is caused by buffering. Operating system can increase the delay if Nevot is run as a time-sharing class process because then it’s not guaranteed to be scheduled at predicted time intervals. In these measurements it varied between 0 and 2.9 milliseconds. Components of the end-to-end delay are illustrated in Table 3.

Table 3. Components of the end-to-end delay

<i>Delay component</i>	<i>Delay in ms</i>
Framing delay	20.0
Processing delay	0.5 – 3.7
HW delay	2.2
Network delay	0.5
OS delay	0 – 2.9
Buffering delay	8.2 – 8.8

An interesting thing was noted in the end-to-end delay behaviour with Nevot. When a new talkspurt is started, the delay is first around 30 ms as shown earlier, but within two seconds it increases by 40 ms and after about 10 seconds it increases by another 20 ms. Similar behaviour was noted in [1]. This was found out to be caused by the Nevot that was receiving packets. The reason is just to avoid buffer underflows in case of late packets.

We removed this feature by modifying the source code. No reduction to sound quality caused by this change was noticed in our informal subjective listening tests. The playout delays in Nevot are calculated with algorithm 1, which was presented in section 2.3. After the correction no additional playout delays are presented.

3.3 Comparison of playout algorithms

In this section we compare the performance of the playout algorithms that were presented in Section 2.3. The performance metric we use to compare different playout algorithms is the average playout delay vs. loss percentage. To evaluate algorithms 1-4, we generate some traces, which include the sender and received timestamps of each packet from a trace. Using Matlab programs we can simulate the different algorithms. Playout delays for each packet are calculated and we can determine if a packet has arrived before its playout time. Thus we are able to calculate the loss percentage and average playout delay for each algorithm with each trace.

3.3.1 Generation of the traces

In order to simulate playout algorithms, we have to generate some traces which illustrate the delays experienced by voice packets. In the setup of this measurement a half-duplex voice connection is established between the two Sun Ultra Enterprise 1 workstations. A 1 kHz sinus signal is supplied by a function generator to the sendind workstation’s microphone input. The signal generator is manually switched on/off to generate talkspurts and silent periods. Both workstations are running Nevot 3.35 and transmitted and received RTP-timestamps are recorded to files using the debugging option of Nevot. Network load was generated using Radcom Prism 200 protocol analyzer. Loads used with different traces are shown in Table 4.

Table 4. Used network loads

Tr.	Traffic description	Frames/s	Bits/frame	Load/Mbps	Length/ packets
1	Small packets	3000-5000	160	3.936-6.560	25830
	Small packets, high load	2000-3500	320	5.184-9.072	25023
2	Large packets, bursty load	100-860	1450	1.163-10.000	24048
3	Variable size pac- kets, bursty load	200-860	160-1450	0.2624-10.000	25337
4					

3.3.2 Comparison of algorithms 1-4

In this section we compare algorithms 1-4 using each of the traces with parameters that gave the best performance in our experiments. For algorithm 1, $\alpha = 0.999$ is used for traces 1 and 2, $\alpha = 0.990$ is used for trace 3, and $\alpha = 0.998002$ is used for trace 4. For algorithm 2, $\alpha = 0.990$ in all traces and for

algorithm 3, $w = 5000$ in all traces. Algorithm 4 uses the same values for α as algorithm 1, and the value for $w = 5000$. Results are shown in Figures 2-5.

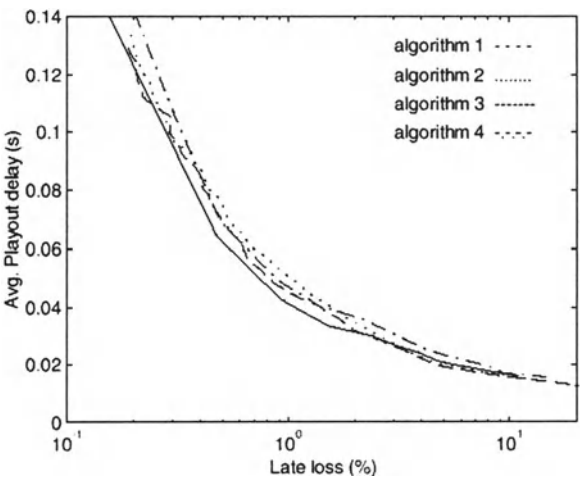


Figure 2. Algorithms 1-4 on trace 1

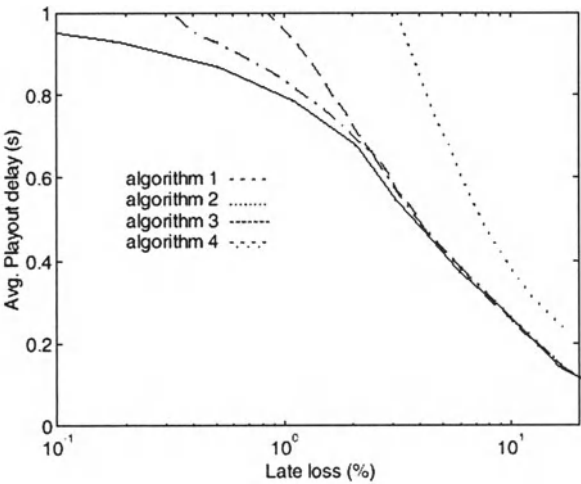


Figure 3. Algorithms 1-4 on trace 2

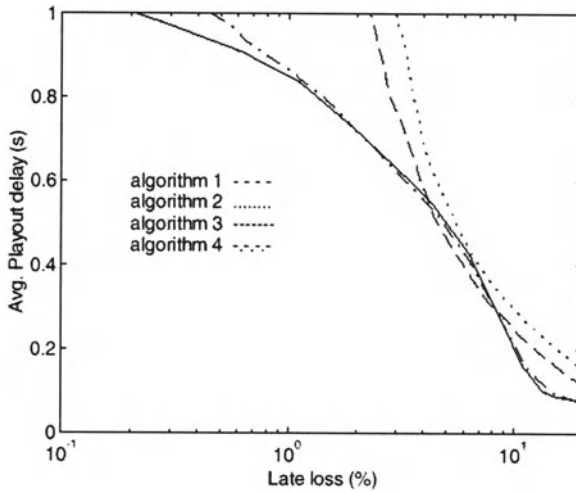


Figure 4. Algorithms 1-4 on trace 3

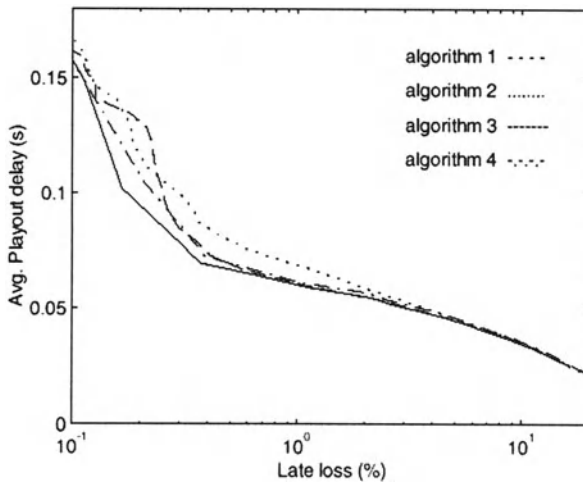


Figure 5. Algorithms 1-4 on trace 4

We can see that algorithm 3 gives the best performance over the other algorithms on all traces except on trace 3, where algorithm 1 gives the best performance at the loss rates of 4-8 %. The differences between the performance of different algorithms seemed to increase with smaller loss rates and greater delay variance. With traces 2 and 3, algorithms 1,3 and 4 gave almost similar performance at loss ratios over 4 %, but when loss ratios decrease, algorithms 3 and 4 perform a lot better.

Logically, algorithm 4 shows performance between algorithms 1 and 3 except with trace 1 where the results are about the same for algorithms 1 and 4. In the higher scale of packet loss, algorithms 1,3 and 4 perform equally well. When packet loss reduces, algorithm 3's and algorithm 4's performances improve compared to algorithm 1.

If we are using 20 ms voice packets then 5000 packets would mean 100 seconds of voice. When we consider that in average 40 % of the time the sender is active, this means that the switch from algorithm 1 to algorithm 3 happens approximately 250 seconds after the beginning of a call. The actual benefit of algorithm 4 is obtained during long calls, with duration over 4 minutes. But compared to algorithms 1 and 2, the use of a better spike detection mode from algorithm 3, improves the presentation from the beginning of a call.

4. CONCLUSIONS

In this paper we concentrated on the delays in an IP voice terminal. We measured the processing delays in a Voice over IP application software using different audio codecs. Even with the heaviest codec, the contribution of processing delay to the end-to-end delay stayed under 4 ms in a 20 ms frame time.

We also studied the end-to-end delay between two Sun Ultra workstations in a situation where the network between the workstations was unloaded and practically all delay was generated in the workstations. The components of the end-to-end delay were measured and a significant part of the delay was found out to be generated in the receiving workstation where the application software starts collecting voice packets to the playout buffer in order to avoid buffer underflows. Other components of the end-to-end delay were framing delay, processing delay, other buffering delays, HW delay in the soundcard and the operating system delay.

In our measurements, the difference in the end-to-end delays between real-time and time-sharing class processes was relatively small, between 0 and 3 ms, but these measurements were performed in the summertime when the workstations were lightly loaded.

Our experiments show that the Sun Ultra platform provides an environment where it is possible to provide bounds on the delays presented in the workstations. This can be accomplished with real-time scheduled processes that are provided by the operating system. If the application software is implemented so that it presents no additional delays and the used network connection is lightly loaded, it is possible to achieve end-to-end delays in the order of 30-40 milliseconds using 20 ms packet size.

We also made comparison of different playout algorithms under different network delay characteristics. The best performance in our simulations was obtained by an algorithm that was based on previous delay history. This algorithm was not real-time implementable as such and therefore we presented a new algorithm that was a combination of this and an existing real-time algorithm. Our algorithm was shown to outperform the other existing real-time algorithms that were compared in our studies. This algorithm gives best performance with calls that last over 250 seconds. During the first 250 seconds, delay statistics are collected and from there on, playout delays are calculated using the delay distribution of previous 5000 packets.

In this paper we simulated playout algorithms using traces that were generated by loading the network with a protocol analyzer. Future work includes simulations by using actual network traces obtained between different geographical locations and comparing the results with those presented in this paper.

ACKNOWLEDGEMENTS

This work was carried out in IPANA-project funded by Tekes and the industrial partners Miratel, Helsinki Telephone Company, Telecom Finland, Tellabs, Nokia Research Center and Nokia Telecommunications.

REFERENCES

- [1] Yletyinen, T. (1997). Quality of voice over IP. *Master's thesis*, Helsinki University of technology, Telecomm tech. <http://keskus.hut.fi/tutkimus/ipana/paperit>.
- [2] Yletyinen, T. and Kantola, R. (1998). Voice packet interarrival jitter over IP switching. *ITS '98*, Brazil.
- [3] Telecommunication Standardization Sector Of ITU. (1993). ITU-T Recommendation G.114. *Technical report*, International Telecommunication Union.
- [4] Jayant, N. (1980). Effects of packet loss on waveform coded speech. *Fifth Int. Conference on Computer Communications*, pp.275-280, Atlanta, GA.
- [5] Montgomery, W. (1983). Techiques for packet voice synchronization. *IEEE Journal on Selected Areas In Communications*, 6(1), pp.1022-1028.
- [6] Alvarez-Cuevas, F., Bertran, M., Oller, F., Selga, J. (1993). Voice synchronization in packet switching networks. *IEEE Networks Magazine*, 7(5), pp.20-25.
- [7] Bolot, J. (1993). End-to-end packet delay and loss behavior in the Internet, *Proceedings of ACM SIGCOMM '93*, pp.289-298, San Francisco, CA.
- [8] Ramjee, R., Kurose, J., Towsley, D., Schulzrinne, H. (1994). Adaptive playout mechanisms for packetized audio applications in wide area networks. *Proc. IEEE Infocom '94*, Montreal, Canada
- [9] Moon, S., Kurose, J., Towsley, D. (1995). Packet Audio Playout Delay Adjustment: Performance Bounds and Algorithms. *Technical Paper*, Dept. of Computer Science, Univ. of Massachusetts at Amherst.
- [10] Schulzrinne, H. (1992). Voice communication across the Internet: A network voice terminal. Univ. of Massachusetts, USA.

ON LOAD BALANCING IN DISTRIBUTED TELECOMMUNICATION SYSTEMS

Per-Oddvar Osland

Department of Telematics

Norwegian University of Science and Technology

peroo@item.ntnu.no

Peder J. Emstad

Department of Telematics

Norwegian University of Science and Technology

peder@item.ntnu.no

Abstract Distributed systems providing telecommunication services, such as IN and TINA, must satisfy strict real-time requirements. Load control and load balancing are two closely connected policies that may be applied to achieve low rejection probability for new calls, maintained response times for accepted calls, and balanced load on available resources. This paper presents requirements and performance measures for dynamic load balancing policies for telecommunication systems, as well as a brief review of the latest research in the field.

1. INTRODUCTION

Systems for provisioning of value added telecommunication services are subject to increasing use. With extensive implementation of mass services such as Number Portability (NP), Virtual Private Networks (VPN), Universal Personal Telecommunications (UPT), and Televoting, almost every phone call will require some processing. In architectures such as the Telecommunication Information Networking Architecture (TINA) and the Intelligent Network (IN), this processing will be performed at dedicated nodes. TINA provides a distributed architecture for service provisioning, and recent publications (Bonnet and Marchese, 1998; Arvidsson et al., 1999; Osland and Emstad, 1999) argue that IN should also be distributed in order to provide dependability and improved performance. Calls made may now be viewed as a flow of service

requests, initiated by users throughout the network, towards the processing nodes. We assume sufficient network capacity, and hence these nodes will be potential bottlenecks in high load situations.

People are used to the telephone as a quick and reliable way to communicate. Frequent call rejection and/or long response times are perceived as indications of severe performance degradation. A major challenge when providing value added telecommunication services will be to meet customers demand for availability and QoS. By QoS we mean minimal rejection of new calls and, for admitted calls, to keep response times at an acceptable level. Optimal resource usage is needed to meet these requirements.

Load balancing and load control policies may be employed to fulfill the above stated demands. A *load control* (LC) policy regulates the admission of new service requests; it attempts to reject as few requests as possible while at the same time assuring sufficiently low response times for admitted requests. In order to optimise overall network performance, the policy should operate at a network level rather than on a node by node basis. A *load balancing* (LB) policy distributes the accepted service requests on the available processing nodes in order to provide fairness in terms of equal expected response-times, and to keep system utilisation up, i.e. avoid situations with some idle nodes while others are heavily loaded.

These policies are closely coupled, and are sometimes considered as two aspects of one policy. LC is a mature and well studied field, whereas LB for telecom systems has not received the same attention until quite recently. The scope of this paper is to render some basic concepts on load balancing, and to indicate cautions to be taken when designing LB policies for distributed telecommunication systems. The objectives of an integrated load control/load balancing policy will be addressed.

The paper is organised as follows: In Section 2. we present some issues to be considered when designing LB policies for telecom systems. Terminology, classification, requirements and performance measures for load balancing policies are addressed in Section 3. A literature review is given in Section 4. before concluding remarks and indications for further work which are found in Section 5.

2. LOAD BALANCING POLICIES FOR DISTRIBUTED TELECOM SYSTEMS

Today we observe a steady convergence between the computer science and telecommunication worlds, where a range of audiovisual services are offered through different networks. Some of these services are real-time

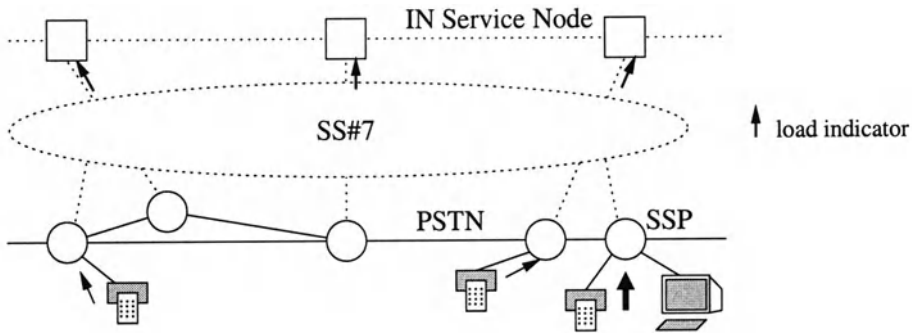


Figure 1 Load balancing improves IN performance.

telecom applications like NP, VPN, and UPT. For systems providing these services, such as IN and TINA, there must be specific requirements to policies for load balancing and control. In this section we present some issues that must be considered when formulating these requirements.

Load balancing and control in distributed systems is a vast and well studied area. As pointed out by Rumeswicz, 1999, the literature is dominated by two views. On the one hand, LB is often used to minimise response times for jobs (Eager et al., 1986; Kunz, 1991). This results in low response times, but if LC is not an issue, it also leads to severe performance degradation in overload situations. In other contributions, in the telecom literature in particular (see e.g. Berger, 1991; Rumsewicz and Smith, 1995), load control has received much attention. Here emphasis has been on keeping system throughput and response times at an appropriate level for different traffic load cases.

An obvious suggestion would be to apply LB results from the distributed systems literature to problems in the telecom world. To understand why this is not immediate, we need to review some important properties of architectures providing value added telecom services.

- As discussed above, most jobs (service requests) in the telecom world are subject to real-time requirements. Therefore emphasis should be on maintaining response times and minimising rejection probability rather than minimising response times and optimising throughput.
- A common situation that arises in distributed operating systems is when nodes receive jobs that can either be processed locally or migrated for remote processing. In a worst case scenario, this could result in a job being migrated several times before finding a node

that accepts it for processing. The situation is somewhat different when dealing with services with strict real-time requirements. In IN and TINA, some nodes receive service request, and forward them to other nodes for processing. In an IN context, the SSPs will schedule jobs, while the Service Nodes will process them (see Figure 1). A service request can be scheduled for a processing node only once. Once scheduled, further migration or job rejection is intolerable.

- A computer network considered is usually a LAN or WAN, while a telecom network, often covering an entire country, is far more widespread. Network capacity is typically higher in the former case, e.g. compare a 10Mb/s WAN with 64 kb/s in Signalling System No 7. Therefore network delays should be taken into account when doing performance evaluation of load balancing policies in telecom networks. This aspect is seldom considered as crucial, and has been treated only by a few publications. Mirchandaney et al., 1989, account for delay when distributing jobs, but assumes negligible delays when distributing load information between the nodes. Kremien and Kramer, 1992, investigate several published load scheduling policies while taking network delays into account.

3. TERMINOLOGY, CLASSIFICATION AND QUANTIFICATION

The proper design of a load balancing policy depends on the target system. In this section we first clarify some terminology, then describe a general classification of LB policies. Furthermore a set of explicit requirements and performance measures for LB policies in distributed telecom systems is presented.

Load distribution is the migration of jobs to a set of nodes for remote processing. *Load scheduling*, more precisely called spatial load scheduling, is the act of selecting a specific node for job transfer. Temporal load scheduling, which means to permute jobs (e.g. according to job size in order to maximise throughput), is not within the scope of this article. The scheduling can be classified as deterministic (e.g. Round Robin) or probabilistic. In the latter case, *scheduling probabilities* indicate the probability for selecting a specific node for job transfer. According to Eager et al., 1986, the purpose of *load sharing* is to obtain high system utilisation, or to never let a node be idle when a job is waiting. There is no restriction on the number of jobs in each node, nor is there a requirement that all jobs should get the same response times. *Load balancing* means to strive for equal load on nodes, which

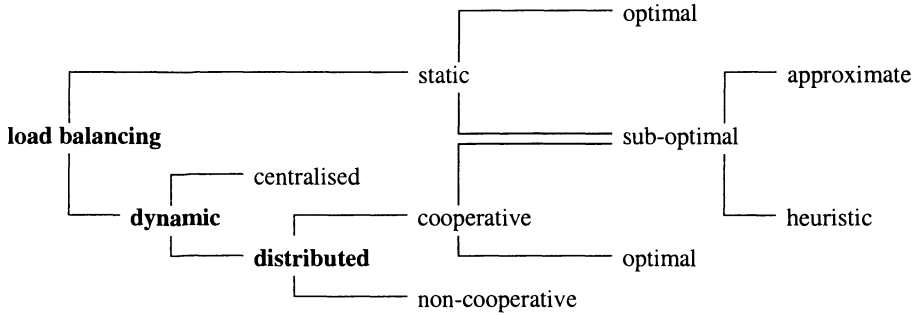


Figure 2 Casavant's classification of load balancing policies

in a homogeneous environment (equal capacity on servers) means equal queue lengths. A node is modelled as a queue plus a server, hence the response time (aka sojourn time) is the sum of queueing and processing time. In a non-homogeneous system we say that the load is balanced when a new job will have the same expected response time, independent of which node it will be sent to. This is equivalent to # of jobs in a node times expected service time being the same for all nodes.

3.1 Classification

Using the classification of Casavant and Kuhl, 1988 (see Figure 2), we learn from the so far sparse literature on LB in telecom systems (Kihl et al., 1999; Rumeswicz, 1999; Arvidsson et al., 1999; Osland and Emstad, 1999), that dynamic, distributed load balancing policies seem to be best suited. A *dynamic* (or adaptive) policy takes system state into account when scheduling new service requests. A *distributed* algorithm resides and acts in several nodes in the network. In particular, a policy is *symmetrically distributed* if the same policy resides in all nodes. In an IN context, the SSPs distribute load on the Service Nodes. Then the LB policy could be placed in the SSPs. A policy is cooperative if it allows nodes to migrate load rather than processing all jobs locally. Since the choice for a scheduling node in IN or TINA is either to migrate or reject a job, this classification is irrelevant for the scope of this article. A thorough classification and qualification of policies is also performed by Kremien and Kramer, 1992, who conclude that dynamic, symmetrically distributed LB policies are the most promising for computer systems.

An adaptive policy requires the distribution of load information from processing nodes to scheduling nodes. In a highly distributed telecom

environment this could be a demanding process, and network delays introduce the problem of out-aged state information. Hence the frequency and amount of information to be exchanged is determined by a trade-off between overhead (transport and processing of state information) and the usefulness of the information.

3.2 Requirements

Most of these requirements are taken from Kremien and Kramer, 1992; Arvidsson et al., 1999; Rumeswicz, 1999.

1. The policy must be simple to implement.
2. The policy must be quick and require little system resources in terms of memory for algorithm construction and run-time cost.
3. The amount and exchange frequency of state information must be moderate enough to only constitute a small fraction of the entire traffic in the system. Policies that require frequent exchange and much system information must be significantly better than less demanding policies.
4. The hit-ratio, i.e. the fraction of correct decisions made by the policy must be better than that of a random policy.
5. Adaptability: The policy must work well for different traffic patterns.
6. Scalability: Change in size of the system should not affect policy performance significantly.
7. Stability: As a result of load balancing, the state in the system should gravitate toward the stable state of the system. Oscillating or diverging behavior indicates poor performance.

The following requirements are particularly important to a load balancing policy for telecom architectures:

1. The primary goal should be to satisfy customers demands, and may be formulated as the following optimisation problem: *Minimise rejection probabilities while maintaining response times at or below a given limit.*
2. Optimise resource utilisation, i.e. maximise utilisation while maintaining response times.
3. Balanced load on processing nodes.

4. Accepted calls must be given priority over fresh calls.
5. Fairness: Unless priority is an issue, different users and services should be treated fairly.

The demand for limited response times implies that processors may have to run at a load lower than the load that maximises throughput.

3.3 Performance measures

To validate a policy, performance measures for the following baseline policies can be compared: *no balancing*, *random*, and *ideal*. To be useful, a dynamic policy must be better than the random policy, and by far outperform a situation where no load balancing is used. The *ideal* policy always “sees” the best solution for every scheduled job. Comparison with an ideal situation serves as a measure for how close a policy is to an unachievable bound. Below we give some general indications for evaluation of LB policies with respect to the requirements stated in Section 3.2.

Response times: Let S be the response time for a job in a node, and s^{limit} the limit that S should not exceed. Since emphasis should be on maintaining S rather than minimising it, $P(S > s^{limit})$ is a more suitable performance measure than $E[S]$. A contribution that supports this choice may be found in Hvasshovd et al., 1995, where requirements for a telecom database are stated. They suggest $s^{limit} = 15\text{ms}$ and require $P(S > s^{limit}) < 5\%$. The following inequalities may be used when comparing with the no balancing and random policies:

$$\frac{P(S > s^{limit} \mid \text{LB})}{P(S > s^{limit} \mid \text{no balancing})} \leq T_1$$

and

$$\frac{P(S > s^{limit} \mid \text{LB})}{P(S > s^{limit} \mid \text{random})} \leq T_2$$

As indicative values, we suggest $T_1 = 0.1$ and $T_2 = 0.5$. Furthermore, the maximum difference in expected response time at the servers must be small, say less than 20%.

Rejection probability: $P(\text{rejection})$ for call attempts:

$$\frac{P(\text{rejection} \mid \text{LB})}{P(\text{rejection} \mid \text{no balancing})} \leq R_1$$

and

$$\frac{P(\text{rejection} \mid \text{LB})}{P(\text{rejection} \mid \text{random})} \leq R_2$$

As indicative values, we suggest $R_1 = 0.1$ and $R_2 = 0.5$.

Hit-ratio: When scheduling a job for a server, we define a correct hit as the scheduling decision made by the ideal policy. As a general requirement the hit-ratio for a LB policy should be between .5 and 1, and must be better than for the no balancing and random policies.

Scalability: A scalable policy works well for systems of different sizes, and is indicated by

$$\frac{\text{performance}(\text{LB}(\text{scalability-parameter A}))}{\text{performance}(\text{LB}(\text{scalability-parameter B}))} = 1 \pm \alpha$$

where $\alpha \ll 1$, and $A \gg B$. Number of nodes, server capacity and call arrival intensity are typical scale parameters. Slight changes in policy settings may be an acceptable compensation for sensitivity to scale.

Fairness: For a mix of services, users requesting the same service should experience the same response time and rejection probability. Every service type should be entitled a fair share of the entire processing capacity. In particular, situations where some dominating services (e.g. Televoting) suffocate other services must be avoided.

Balanced load: The load on the processing nodes must be balanced. If utilisation differs significantly from node to node, higher utilisation at the fastest servers should show.

4. LOAD BALANCING: A LITERATURE REVIEW

Dynamic load balancing policies comprise two main activities; information exchange and decision making. The policies reviewed in this section mainly differ in the way these activities are implemented.

Information exchange strategies determine how to store, exchange, and update information such that the scheduling nodes keep a local view of the system's state. Queue length is a representative and much used metric for system state. Information exchange strategies can be characterised as either periodic or state-driven.

The *decision making* part of the policy indicates to which node a job should be scheduled for processing, or if it should be rejected. A typical

strategy is adaptive, probabilistic scheduling, where scheduling probabilities are weighted in order to reflect the load status in the system. The scheduling probabilities indicate the probability for selecting a specific node for job transfer. In an adaptive policy the scheduling probabilities are updated regularly as new state information appears.

Kihl et al., 1999, use no explicit load information exchange when evaluating load balancing algorithms for TINA networks but, rather, count the number of jobs rejected when sent for remote processing. This simplifies the policies, and also makes them well suited for scalable networks. However, state information is exchanged only at high load since there will be no rejections when the load is low. The rejection counts are used to define scheduling probabilities in two policies. These are compared with the random and ideal baseline policies. Tests show that it is difficult to find a policy that works well for a variety of traffic situations. As an improvement, Kihl et al recommend adding explicit load information to the TINA protocol.

A neat way to obtain additional load information (still implicit) is presented by Hosein, 1999, who considers overload control for an IN database. He lets the response time of a job at a processing node indicate the load on the node. This works both for high and low load situations, and the policy shows good performance under varying server intensity. However, if network delay is significant, this method may provide inaccurate information.

Arvidsson et al., 1999, propose to use agent technology for the exchange of load information. They use a 'Market-Based Control' approach to distribute IN processing capacity, represented by tokens. The information dissemination is periodic; every 10 seconds an auction is held where agents representing the SCPs and SSPs sell and buy tokens. By letting the SCPs offer available processing capacity in terms of tokens, LC and LB is achieved. The agents acting for the SSPs express the SSPs' need for processing capacity by placing bids in the auction. Through mechanisms in the auction the following goals are achieved: SCP utilisation close to .9, maximum network profit, minimum response delays, fairness to services and users, and equally balanced load. A great advantage is that there is no need for synchronisation of the parts in the auction. The strategy turns out to work well for different traffic load scenarios. Still it remains to estimate cost for holding the auction and for the activity of the agents (signalling between agents, extra memory and processing required) in order to justify the policy.

Up to this date, the most comprehensive policy is probably presented by Rumeswicz, 1999. He models a call as composed of several jobs that can be processed at different nodes, and gives preference to calls which

have received a significant amount of processing over recently commencing calls. The policy is scalable because it makes no a priori assumption about processor capacity or placement of functionality on processors. It also works well for different traffic mixes, hence it is adaptive. When it comes to information exchange, the processing nodes distribute a relative load indication, each node having its own frequency of emitting load information. The policy does not require synchronisation of controls taken, it attempts to maximise the number of calls completed, and it is simple to implement.

In a previous paper on performance evaluation of load scheduling policies for IN (Osland and Emstad, 1999), we describe a policy where an SSP explicitly asks for load information from the Service Nodes. This makes the load distribution event-driven and synchronises it in the sense that Service Nodes distribute their load simultaneously. Synchronised information dissemination makes the policy suitable for analytical performance evaluation with network delays taken into account. Upon reception of state information, this policy compares the expected response time in the Service Nodes with predefined threshold values. Scheduling and rejection probabilities are then set in order to minimise rejection probability while keeping response times below a given limit. Analytical and simulation results show that this policy satisfies its goals, and the utilisation is rather good compared with an optimal policy. A weak point is that the policy does not scale well when system size changes, and that frequent update of state information is assumed.

5. CONCLUSIONS AND FUTURE WORK

In systems providing telecommunication services, such as IN and TINA, load balancing and control are two closely connected policies used to satisfy critical requirements such as maintained response times, low rejection probability and balanced resource usage. In this paper we have presented requirements and performance measures for dynamic load balancing policies for telecommunication systems. A brief review of the latest publications in the field has also been given. The review features scalable, adaptive, and efficient policies. Researchers do not agree if emphasis should be on maintaining response time and minimising rejection probability, or to maintain goodput while minimising response times. We support the former, which means optimal QoS from a customers point of view. The latter means optimal resource usage from a operator's point of view.

A topic for further work is to compare the algorithms to show their pros and cons with respect to relevant performance measures. Work is

in progress to formulate and study more efficient policies that take the problem of network delays into account.

Acknowledgements

We would like to thank Professor Don Towsley of University of Massachusetts at Amherst for his helpful comments and suggestions.

References

- Arvidsson, Å., Jennings, B., Angelin, L., and Svennson, M. (1999). On the use of agent technology for in load control. In Key, P. and Smith, D., editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th International Teletraffic Congress*, volume 3b, pages 1093–1104, Edinburgh, Scotland. IEE, Elsevier.
- Berger, A. (1991). Comparison of call gapping and percent blocking for overload control in distributed switching systems and telecommunications networks. *IEEE Transactions on Communications*, COM-39, 4:574–580.
- Bonnet, M. and Marchese, P. (1998). A model of Intelligent Network distributed computing for Number Portability. In *Proceedings of the 5th International Conference on Intelligence in Networks*, pages 378 – 383, Bordeaux, France.
- Casavant, T. L. and Kuhl, J. G. (1988). A taxonomy of scheduling general-purpose distributed computing systems. *IEEE Transactions on Software Engineering*, 14(2):141–154.
- Eager, D. L., Lazowska, E. D., and Zahorjan, J. (1986). Adaptive Load Sharing in Homogeneous Distributed Systems. *IEEE Transactions on Software Engineering*, 12(5):662–675.
- Hosein, P. (1999). Overload control for real-time telecommunication databases. In Key, P. and Smith, D., editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th International Teletraffic Congress*, volume 3a, pages 263–272, Edinburgh, Scotland. IEE, Elsevier.
- Hvasshovd, S.-O., Torbjørnsen, Ø., Bratsberg, S., and Holager, P. (1995). The ClustRa Telecom Database: High Availability, High Throughput, and Real-Time Response. In *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB'95)*, pages 469–477, Zurich, Switzerland.
- Kihl, M., Widell, N., and Nyberg, C. (1999). Load balancing algorithms for TINA networks. In Key, P. and Smith, D., editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th Interna-*

- tional Teletraffic Congress*, volume 3b, pages 999–1008, Edinburgh, Scotland. IEE, Elsevier.
- Kremien, O. and Kramer, J. (1992). Methodical analysis of adaptive load sharing algorithms. *IEEE Transactions on Parallel and Distributed Systems*, 3(6):747–760.
- Kunz, T. (1991). The influence of different workload descriptions on a heuristic load balancing scheme. *IEEE Transactions on Software Engineering*, 17(7):725–730.
- Mirchandaney, R., Towsley, D., and Stankovic, J. A. (1989). Analysis of the effects of delays on load sharing. *IEEE Transactions on Computers*, 38(11):1513–1525.
- Osland, P.-O. and Emstad, P. J. (1999). Performance evaluation of load scheduling policies for Intelligent Networks. In Key, P. and Smith, D., editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th International Teletraffic Congress*, volume 3a, pages 283–292, Edinburgh, Scotland. IEE, Elsevier.
- Rumeswicz, M. (1999). Load control and load sharing for heterogenous distributed systems. In Key, P. and Smith, D., editors, *Teletraffic Engineering in a Competitive World. Proceedings of the 16th International Teletraffic Congress*, volume 3b, pages 1083–1092, Edinburgh, Scotland. IEE, Elsevier.
- Rumsewicz, M. P. and Smith, D. E. (1995). A comparison of SS7 congestion control options during mass call-in situations. *IEEE/ACM Transactions on Networking*, 3, 1:1–9.

ENABLING FACTORS FOR IN SERVICE PROVIDER ACCESS

A.Jøsang*, M.G.L.Bijnagte** and G.Kleinhuis**

* The Norwegian University of Science and Technology, N-7149 Trondheim, Norway

** KPN Research, NL-2264 XZ Leidschendam, The Netherlands

Key words: Intelligent networks, Service provider access, INAP, SS7, TCP/IP

Abstract: The need to offer IN services across network boundaries is expected to come as a result of the ongoing deregulation, globalisation of telecom markets and the increased mobility of users. This will lead to a pressure for a clearer separation between IN service providers and network operators allowing third party service providers open access to public networks. Some operators may want to resist this development on the grounds of avoiding competition, but we claim that maintaining this view is counterproductive in a larger perspective, and that all parties will benefit from open service provider access to public networks. This paper points out that legislation and standardisation are essential factors for making public networks open for third party IN service providers, and proposes to use the TCP/IP protocol stack for the IN service provider interface.

1. INTRODUCTION

Telephony services depend on service logic in several parts of the network. Even a basic call requires the prefix to be analysed in the originating exchange and the local subscriber identification to be performed in the terminating exchange. When two operators cover separate geographical areas, the mutual benefit of this type of cooperation is obvious, and interconnection agreements are relatively easy to negotiate. On the other hand, if operators compete in offering services to the same subscribers, the mutual benefit is less obvious, resulting in operators being reluctant to

cooperate. This difference is for example reflected by the well defined standards for GSM roaming where mobile network operators co-operate in order to provide a seamless radio interface to their subscribers, in contrast to the almost non-existing standards for providing competing services in the same access network or geographical area.

The main characteristic of the intelligent network (IN) concept is that the logic for controlling calls and call related services is separated from the physical switching and set-up of calls. For the provision of IN services it is natural to distinguish between the network operators and the service providers although they usually are one single organisation in today's networks. The public network operator (PNO) owns and operates switches, transmission lines and subscriber access networks. Service providers own and operate data and logic for determining routing and for handling call related services, as well as supporting resources such as for example speech recognition equipment or service management tools. An organisation can play the role of PNO, of IN service provider, or both. A service provider will be called *third party service provider* (3PSP) when offering services through a public network owned by a different organisation.

A company that is both PNO and IN service provider would intuitively be reluctant to grant 3PSPs access to its network because it would create competition in the own network. We claim that companies maintaining this view are seeing the telecom market in a too narrow perspective. Third party service provider access to public networks will enhance the quality of telecom services in general and thereby result in increased service usage and market growth, ultimately leading to increased revenue for the network operators and service providers themselves.

In existing networks IN is usually integrated with the switching equipment of the network operator. Even when telecom companies are split into separate legal and commercial entities where the IN service provider subsidiary is different from the PNO subsidiary, it is usually only in the name. In reality, the PNO subsidiary still operates the equipment that delivers the actual IN services, whereas the service provider subsidiary is only a marketing company that in fact hires all the telecom resources needed for providing IN services from the PNO subsidiary. By opening up public networks for IN service provision, the separation has to be real, i.e. there must be a physical interface to the public network.

This paper will discuss the necessary factors for making public networks open to third party service providers. Regulation is seen as essential to overcome the competition deadlock and to provide a stable and predictable environment for new service providers. Secondly, standardisation will be essential to make open access practical. In particular we propose to let the TCP/IP protocol stack be used instead of the SS7 protocol stack in the

interface between the PNO and the 3PSP. Finally, open IN service provider access will result in more complex networks. This will require advanced technology and an infrastructure that is sufficiently mature for mastering for example service interaction and security problems.

2. SEEING THE BENEFITS OF OPEN SERVICE PROVIDER ACCESS

Competition is a stimulator for innovation, increased efficiency and better services. However, this may be difficult to see for companies that try to increase their profit in the short term. In many cases it depends on the time perspective a company uses, and it is important to realise that rational behaviour in a short perspective can be counterproductive in a larger perspective and vice versa.

Today's telecommunication networks have not been designed with open service provider access in mind. Their structure reflects a monolithic enterprise approach and their ability to serve user mobility and global integration is limited. IN is partly a solution to this problem. However, if IN services cannot be provided across network boundaries, their potential will be limited. From a service quality point of view there are several aspects to consider such as for example geographical coverage, availability at any time and user friendliness, and an open interface at the service provider level will make it much easier to handle these issues in a satisfactory way.

Users should be able to choose between (IN) service providers, so that they no longer are limited to those services that their PNO can offer but are also able to use services from a 3PSP. The services offered by the 3PSP will broaden the service spectrum by offering different services or equivalent services with different quality and price. By opening up public networks for IN service provider access the incumbent PNOs will face competition in an area where they have been in a monopoly situation. However uncomfortable this initially may be, the 3PSPs will become potentially large traffic generators, so that a reduced IN service market share can be compensated by increased traffic.

As soon as one PNO has opened up for IN service provider access, all the other PNOs in the same market will be forced to do the same. A PNO that resists opening up its network will avoid competition in its own network, but will run the risk of losing subscribers, and will at the same time lose 3PSP generated traffic to competing PNOs. The most innovative IN services will only be available through PNOs that grant 3PSPs access to their networks. It will thus be dangerous to come too late with a solution for IN service provider access.

3. THE REGULATORY PRESSURE

The cooperation deadlock caused by competition can only be overcome by regulation such as for example the Directive on Interconnection [1] of the European Union which has as purpose of ensuring universal service and interoperability through the principles of Open Network Provision (ONP), and ultimately of serving the objective of economic and social cohesion and territorial equity.

The EU directive states that “*fixed and mobile operators with significant market power must meet all reasonable requests for access to their network ...*” (Art.4.2). The term “reasonable” means that access should be granted if it is required for offering a service, if it is technically possible, and if it does not threaten the network integrity. Guidelines for the operational aspects of network integrity are given in the recommendations [2] published by the European Committee for Telecommunications Regulatory Affairs.

Furthermore, the EU directive states that interconnection agreements between the involved parties should be non-discriminating, meaning that no discrimination in own use and third party use is allowed, and that the same conditions shall be offered to all the parties asking for interconnection (Art.6.a). Interconnection agreements shall also be transparent, meaning that the agreements must be open and made available for all interested parties (Art.6.c). Finally, interconnection agreements shall be cost-oriented, meaning that the interconnection tariffs must be based on the associated costs of offering the interconnection (Art.7.2).

Finally, the EU directive states that adherence to relevant technical standards may be made compulsory (Art.13). On this background, cost-orientation would imply that the additional cost for service providers when connecting to a network that deviates from existing standards has to be covered by the network operator. Non-standard solutions would in addition break the principle of non-discrimination because it for example would favour interconnecting equipment from the vendor that delivered the network equipment. In the absence of relevant standards, the principles of cost-orientation and non-discrimination are severely weakened, and regulation such as the EU directive alone will not be sufficient to give 3PSPs access to public networks. The existence of relevant standards is thus crucial, and this will be discussed below.

4. POSSIBLE ACCESS INTERFACES

All network elements in an IN architecture can in principle be subject to 3PSP access, but some are more suitable and natural points of

interconnection than others. Figure 1 illustrates a typical single operator/service provider IN architecture with its principal elements.

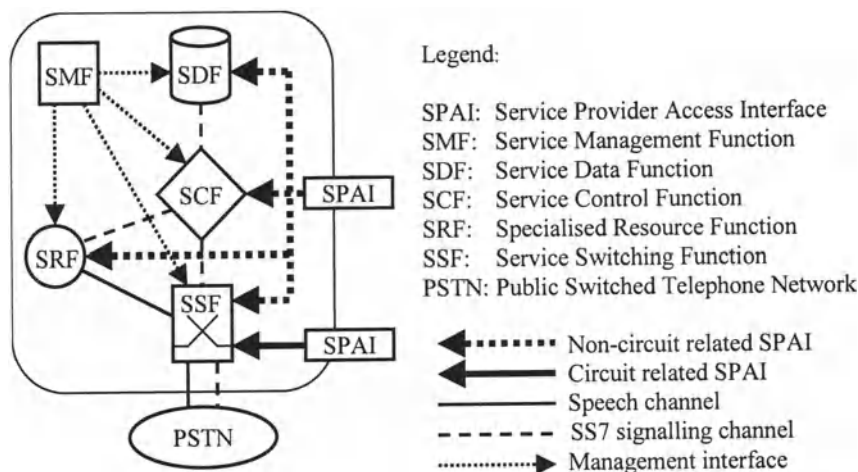


Figure 1 Single operator IN architecture with possible SPAIs.

The required access point will in general depend on the provided service. At a high level the service types can be divided into three categories:

1. **Call control services:** The 3PSP provides the IN services itself. This requires control over the call by the 3PSP. In this case the 3PSP operates its own SCF and requires access either to the SSF or the SCF in the PNO domain.
2. **Support services:** The 3PSP can have a supporting role for the delivery of IN services, by for example providing user information in which case access to the SCF or the SDF is required, or by offering specialised resources like for example speech recognition in which case access to the SCF is required.
3. **Management services:** The 3PSP can manage services and service data on network elements in other domains, such as for example distributing or replicating data to SDFs owned by different PNOs in which case the 3PSP operates a SMF and requires access to SDFs (either directly or via the SMF in the PNO domain). The 3PSP can also have a supporting management role such as for example number portability administration, in which case the IN service provider needs an interface to the portability administrator for the communication of ported numbers.

In the next section, we give a brief overview of the available standards for interfaces offering call control services and discuss their suitability for interconnecting different domains at the IN level.

5. STANDARDS

Intelligent Networks are standardised in phases, called capability sets (CS). These capability sets are indicated with CS1 [3], CS2 [4], etc. CS2 that has recently been finished (1998) –represents a superset of the functionality standardised in CS1 (1995) and is backward compatible to CS1. Within a standard various IN interfaces are standardised. Below the relevant interface standards required for call control services will be discussed.

5.1 Interface standards for call control services

Different options are possible for offering call control across network boundaries, as illustrated in Figure 2.

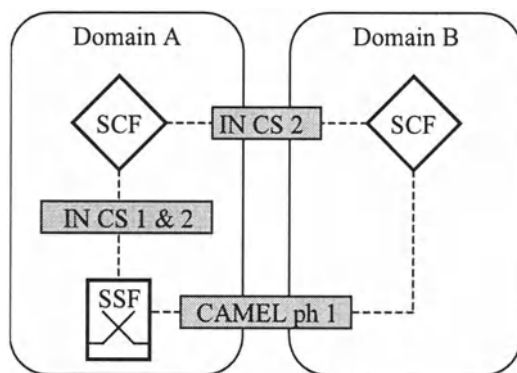


Figure 2. Interface standards for call control services.

For the interfaces required by call control services, the relevant interface standards are IN CS1 SSF-SCF interface, the IN CS2 SCF-SCF interface, and the CAMEL phase 1 interface.

5.1.1 IN CS1&2 for SSF-SCF interdomain connection

The SSF-SCF interface standardised in CS1 [3,5] and elaborated in CS2 [4,6] allows an SCF of the IN platform to have control over the call. This interface also supports charging and user interaction functionality. Implementations of CS1 SSF-SCF are already available in public networks. Since the SSF-SCF interface is standardised for intra-networking, problems arise when using this standard over network boundaries. This standard includes a number of options that would cause obvious problems when used for inter-networking and does not address network integrity such as

authentication and screening functions. Some INAP (Intelligent Network Application Protocol) operations can cause significant problems to the network integrity. For instance (with issuing the `callGap` operation) the SCF can instruct the SSFs to block the sending of IN queries which could be unacceptable. The same applies for the operation `activate-ServiceFiltering` that allows an SCF to instruct the SSFs to filter on specific `serviceKeys`. Also the ability of (re-) setting counters and timers can cause problems. Therefore, at this moment the network integrity is not maintained if this standard is used for inter-networking.

5.1.2 CAMEL phase 1 for SSF-SCF interdomain connection

The CAMEL (Customised Application for Mobile network Enhanced Logic) interface [7] offers a way of interconnecting an SCF and an SSF in a mobile environment. The purpose of CAMEL is to make services from the home domain available in another domain. Phase 1 of this interface is a subset of the existing IN CS1 standard, with some mobile specific parameter extensions. Products for phase 1 have recently arrived on the market. Due to the limited functionality of phase 1, the network integrity is less a problem since powerful operations, like call gapping, and operations for (re-)setting counters and timers are excluded. The use of the CAMEL phase 1 interface for third party service provider access has some drawbacks. In phase 1 no charging and user interaction capabilities are available. CAMEL offers limited functionality compared to the current implementations in fixed networks. This results in discriminatory conditions. Therefore, using CAMEL phase 1 for third party service provider access will probably not meet the 3PSP's (and regulatory) requirements. In addition, implementation difficulties will arise when integrating this standard with its mobile specific parameter extensions in an fixed IN network.

5.1.3 IN CS2 for SCF-SCF interdomain connection

The SCF-SCF interface is standardised within CS2 [4,6] for distributed service execution and can be used over network boundaries. The communication is basically between two co-operating 'service logics' in two SCFs. The SCF triggered by an SSF, called the 'controlling' SCF, initiates the communication and requests a 'supporting' SCF to perform some necessary actions, e.g. determine a routing address for a call or determine the call charge. Products with implementations of CS2 standards are not expected before the year 2000. Authentication procedures which are crucial for maintaining network integrity are specified in the CS2 SCF-SCF interface. In principle, the SCF-SCF interface can be used for

interconnecting third party service providers, but the 3PSP will not directly control the call. The service logic in the controlling SCF has to map the information received from the supporting SCF into the operations instructing the SSF how to process the call. With the disadvantage that for almost every new service offered by a third party service provider specific service logic is needed in the SCF of the PNO. This dependence of the 3PSP on the PNO has consequences for e.g. the time to market of its services and will therefore not meet the 3PSP's requirements.

6. ENABLING TECHNOLOGIES

As we saw in the previous section, the current available standards are not adequate to meet the requirements of offering adequate functionality for call control and maintaining the network integrity. In order to fill this gap, we will describe additions to standards and propose technical solutions for service provider access for call control services. Primarily, additions to the current standards are required. This includes:

- The possibility to perform screening actions on the messages exchanged between the two parties. For instance per traffic relationship between an 3PSP and an PNO a screening profile should be agreed in which the allowed operations, parameters and parameter ranges are negotiated. According to this profile the screening actions will be performed.
- The mapping of network specific options within an PNO or 3PSP domain into bi-lateral agreed formats.
- The possibility to measure transaction flows between an entity in the PNO domain and an entity in the 3PSP domain. For accounting purposes information on traffic flows between an 3PSP and PNO are needed. A large number of 3PSPs may be interconnected to the PNO's IN platform, and therefore capacity has to be reserved. In order to be able to guarantee the reserved capacity, the thresholds for the volumes of traffic negotiated must be checked.

Secondly, the networks must be resistant against malicious attacks, either by external parties, or by one party trying to (whether on purpose or not) abuse the otherwise legal access to another party's network.

The problem of providing secure transactions across domain boundaries is one of the most challenging problems of distributed systems because security ultimately must be based on a common security policy that defines e.g. who is authorised and what is allowed. If different domains cannot agree on a single security policy the interaction must be subject to special agreements between domains on a pair to pair basis. In case of transnational

service provision, problems related to international legal disputes must also be addressed.

The security aspects needed for domain interconnection include:

- **Data integrity:** Only authorised agents shall be able to create, delete or modify messages. Data integrity can be obtained by message authentication codes, by encryption, by authentication or by digital signatures.
- **Confidentiality:** Only authorised agents shall be able to read messages. Confidentiality is implemented by encryption.
- **Authentication:** The recipient of a message shall be able to verify the identity of the sender. Authentication is implemented by the means of protocols consisting of a sequence of messages.
- **Non-repudiation:** The sender of a message shall not be able to deny having sent the message. In the same way, the recipient of a message shall not be able to deny having received the message. Non-repudiation is implemented by using digital signatures.
- **Accountability:** It shall be possible to trace all actions to identified agents, and agents shall be made responsible for their actions.

Authentication and non-repudiation both address the same basic problem of verifying identity. In case the two interworking parties trust each other, authentication is sufficient. In case the parties do not trust each other, a third party mediator must be able to verify the identity of the sender, and that can only be done by non-repudiation based on digital signatures. Non-repudiation is thus stronger than authentication.

In order to provide the missing functionality and security requirements, we propose the introduction of an interworking function where the functionality described above is located. Secondly we propose to use the TCP/IP protocol stack instead of SS7 as the bearer of INAP. Finally we propose to use the Secure Socket Layer protocol (SSL) [8] or its standardised version the Transport Layer Security Protocol (TLS) [9], between TCP/IP and INAP in order to obtain a secure channel between domains. These points will be described separately below.

6.1 The interworking function

The *InterWorking Function* (IWF) acts as a gateway function between the PNO's and the 3PSP's domains. Functionality missing in the standards such as screening, mapping and measuring functions are provided by this IWF. In addition the IWF should perform authentication of the entities involved (see section 6.3 below). Figure 3 illustrates how two domains interact via the interworking function.

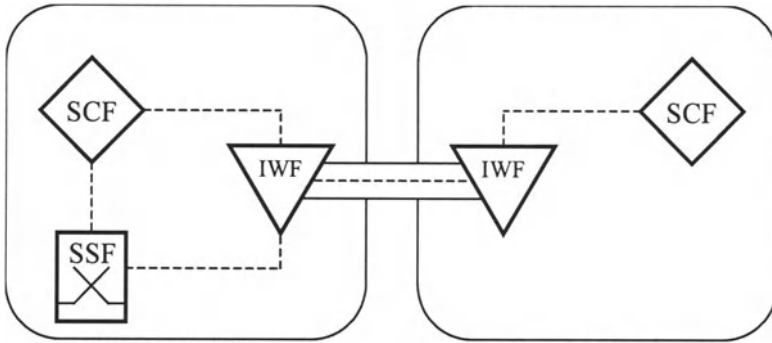


Figure 3. The positioning of the interworking function.

The interworking function is functionally located between the SSF in the 3PSP domain and the SCF in the 3PSP domain. This concept of an interworking function can be used as a gateway both in case of interconnecting between the SSF in the PNO domain and the SCF in the 3PSP domain, as in case of interconnecting between the SCF in the PNO domain and the SCF in the 3PSP domain. Since both the 3PSP and the PNO must be able to perform screening, mapping and measuring, a IWF in both domains is needed. At the physical level the implementation of this function depends on the PNO's and 3PSP's equipment, the interworking function can in principle be collocated in the SCF, as well as in the SSF. A standalone solution is also possible, when placing the IWF in a separate physical unit.

6.2 Using TCP/IP instead of SS7

Presently, almost all IN implementations are using INAP running over Signalling System No.7 (SS7) [10,11,12]. Using SS7 for service provider access imposes several problems. Primarily, the usage of platforms that implement the SS7 protocol stack have been limited to pure telecom applications and remain relatively expensive compared to for example TCP/IP based platforms. Secondly, the TC protocol [10] that is part of SS7 provides powerful functions that potentially could cause severe damage to a network if used inappropriately, and would make network operators more reluctant to open service provider access. Finally, using SS7 would be at odds with the trend of integrating data and telecommunications networks because SS7 is limited to the more traditional telecommunication networks.

We will see more and more applications based on TCP/IP, and the product spectrum based on this technology is much broader and have lower prices than corresponding SS7 based products. A way forward to enable third party service provider access is therefore to use INAP on top of TCP/IP

between network operators and service providers. Figure 4 illustrates the proposed protocol architecture for open IN service provider access, and compares it with the traditional SS7 protocol architecture as well as with the OSI reference model. In addition to the TCP and IP layers, the proposed architecture includes a security layer that will be described in the next section.

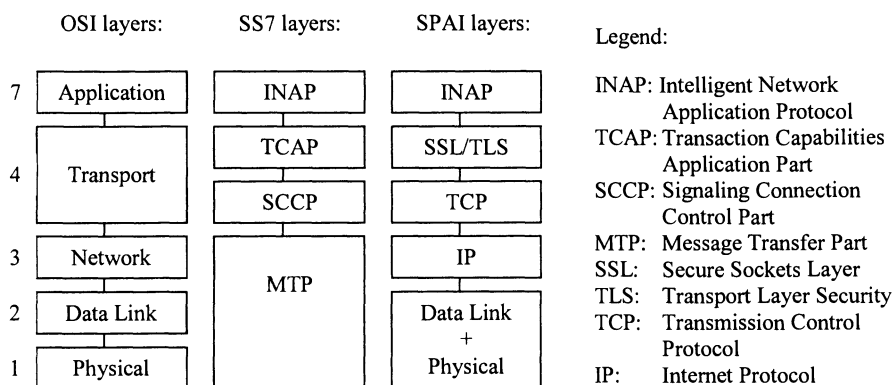


Figure 4. Proposed protocol architecture for IN service provider access.

6.3 The secure channel

Authentication of messages can be obtained by establishing a secure channel that is transparent to the service logic. If required, this can also give data integrity and confidentiality. The secure channel protocol must be standardised to satisfy the non-discrimination requirement. The Secure Socket Layer protocol (SSL) [8] that is already implemented in many products and its standardised version, the Transport Layer Security protocol (TLS) [9] will satisfy these requirements. TLS, which is effectively SSL 3.1, is standardised by the Internet Engineering Task force. TLS will form a protocol layer between INAP and TCP/IP in order to provide the necessary security functions.

Competition can be an incitement to act selfishly, and is therefore likely to create distrust. Regulation alone is not enough to verify that the parties fulfil their obligations, and security mechanisms can be used for this purpose. Security mechanisms can be described as a method to propagate trust from where it exists to where it is needed. Something therefore has to be trusted in the first place.

Two companies that trust each other can exchange keys and establish a secure interconnection without referring to any third party. If on the other hand the parties do not trust each other, the domains must at least agree on a

common trusted third party (TTP) that is trusted to issue certified cryptographic keys. Using these keys, the parties can establish secure channels between each other in order to provide IN services across domain boundaries. This is illustrated in Figure 5 where the trusted third party function (TTPF) is used to distribute certified keys. Based on certificates from the TTPF, the two domains are able to set up a secure channel and interwork to provide IN services.

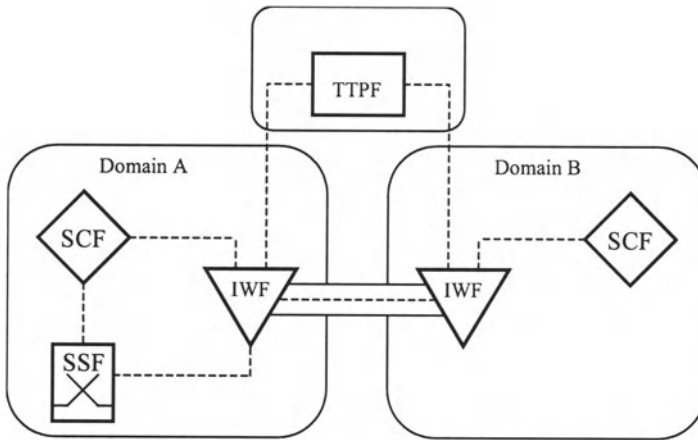


Figure 5. Interworking with a Trusted Third Party Function.

In the framework of public key cryptography, a certificate is a digital signature on an entity's public key, issued by a TTP, guaranteeing the binding between the public key and the identity of its owner. In that way, a 3PSP can use its certified public key to sign commands sent to the PNO and thereby provide the PNO the ability to perform authentication and to assure the PNO that the commands can be traced and not subsequently be repudiated. The X.509 standard [13] for exchanging public key certificates is already used by commercial TTPs, and is the natural choice for establishing a framework for open IN service provider access.

A domain interworking scheme including a trusted third party has already been established in the USA in connection with the regional local number portability. In order to have a fair and equitable solution, the industry has adopted regional neutral administrators, called NPAC (Number Portability Administration Center) that must not be in business as a network operator. There are 7 NPACs in the USA, one per regional Bell operating company area. The role of the NPAC is to receive numbering porting requests from various local network operators through a standard formatted interface called the NPAC Interface Requirement Document [14]. The NPAC coordinates the number exchange in an automated and timely fashion between the

network operators, and has the responsibility to maintain a master database for the assigned region.

7. MASTERING THE INCREASED COMPLEXITY

By allowing telecom users to buy services from different operators and service providers the overall complexity increases. By considering a normal two-party call in a traditional single operator environment there is always a single point of control in each side of the call so that for example feature interaction problems can be controlled to a certain degree. In a multi service provider environment the level of feature interaction control that any operator or service provider can offer is drastically reduced unless a solution for service integration can be found.

Reliable billing is essential for any player in the telecom operator and service market. In an environment with multiple service providers, operators and service providers must be able to log and agree on what network resources have been used by whom. Billing records will have to be collected from many different sources before being presented to the end user, and internal billing between operators and service providers must be done accordingly.

In a single operator environment users had a single point of contact for requests and complaints. Unless operators and service providers can agree on a co-ordinated customer interface for reporting technical problems, the users risk being the victims of a responsibility vacuum.

Handling the problems mentioned above could in fact become separate commercial activities called for example service integrator or service mediator.

8. CONCLUSION

Considering the way in which telecommunication networks have developed in the past 100 years where services and networks have always been integrated third party service provider access does not come naturally. Although technically challenging, it is highly desirable in order to increase competition and remove old monopolistic barriers. This paper has discussed some of the enabling factors for this development. Services can be divided into different categories such as management, support services and call control services, where this paper has focused on the latter. The major problem when interconnecting different domains is to maintain network integrity. For this purpose we have proposed a technical solution which uses

the TCP/IP protocol architecture instead of SS7 for INAP, and which includes a secure protocol layer between the INAP and TCP/IP protocol layers that can be used to establish secure channels between domains. In addition we have proposed an interworking function that acts as a gateway between domains. Apart from these technical solutions provisions need to be made by the public network operators and the third party service providers in order to manage the increased complexity e.g. billing and feature interaction.

REFERENCES

1. Directive 97/33/EC of the European Parliament and of the Council on interconnection in telecommunications with regard to ensuring universal service and interoperability through application of the principle of open network provision (ONP). The European Commission, 1997.
2. Recommendation Rec(98)01 - Network integrity, European Conference Of Postal and Telecommunications Administrations/European Committee for Telecommunications Regulatory Affairs (CEPT/ECTRA), 1998.
3. Recommendation series Q.1210 - Intelligent network capability set 1 (CS-1), International Telecommunication Union, Telecommunication Standardization Sector (ITU-T), 1995.
4. Recommendation series Q.1220 - Intelligent network capability set 2 (CS-2), International Telecommunication Union, Telecommunication Standardization Sector (ITU-T), 1998.
5. ETS 300 374 - Intelligent Network (IN); Intelligent Network Capability Set 1 (CS1); Core Intelligent Network Application Protocol (INAP); Part 1: Protocol specification. European Telecommunications Standards Institute (ETSI), 1994.
6. ETS 301 140 - Intelligent Network (IN); Intelligent Network Capability Set 2 (CS2); Part 1: Protocol specification. European Telecommunications Standards Institute (ETSI), 1999.
7. ETS301 152-1- Intelligent Network (IN); Intelligent Network Capability Set 1 Extension; Intelligent Network Application Protocol (INAP); Part 1: Protocol specification for CAMEL Phase 1. European Telecommunications Standards Institute (ETSI), 1997.
8. The SSL 3.0 Protocol, Netscape Communications Corp. Feb.9, 1995.
9. The TLS Protocol, RFC2246, Internet Engineering Task Force, January 1999.
10. Recommendation Q.773- Specification of Signalling System No.7; Transaction Capabilities formats and coding, International Telecommunications Union, Telecommunication Standardization Sector (ITU-T), 1993.
11. Recommendation Q.711 to Q.714 and Q.716 - Specification of Signalling System No.7; Signalling Connection Control Part (SCCP), International Telecommunications Union, Telecommunication Standardization Sector (ITU-T), 1993.
12. Recommendation Q.701 to Q.708 - Specification of Signalling System No.7; Message Transfer Part (MTP), International Telecommunications Union, Telecommunication Standardization Sector (ITU-T), 1993.
13. Recommendation X.509 - The Directory Authentication Framework, International Telecommunications Union, Telecommunication Standardization Sector (ITU-T), 1993.
14. NPAC SMS Interoperable Interface Specification, NANC version 1.7. Illinois Commerce Commission, 1997.

AN EXPERIMENTAL ENVIRONMENT FOR DISTRIBUTED INTELLIGENT NETWORK SERVICES

Bilhanan Silverajan, Jarmo Harju

*Dept. of Information Technology, Tampere University of Technology, P.O. Box 553, FIN-33101
Tampere, Finland*

{bilhanan | harju}@cs.tut.fi

Keywords: IN Service Creation Frameworks, Distributed Component-based Systems,
Design Patterns, CORBA

Abstract: This paper discusses an experimental lightweight object oriented environment aimed towards the rapid prototyping, testing and deploying of distributed IN services using CORBA. Well-known design patterns have been adapted to implement as well as interwork with CORBA enabled components. Synchronous calls are used for CORBA object requests, while reliable asynchronous delivery such as that needed for all protocol specific transmission is provided via a minimal core subset of an event-based framework known as OVOPS. Thus, no additional CORBA service for messaging or event-handling is needed. The described environment will ensure the consistent creation of IN services as well as needed protocols through reusable object-oriented components communicating over a heterogeneous network. The IN services created would become independent of the underlying protocol which is used to communicate with the other service entities across the network, providing the advantage of being able to run the service over multiple protocols by isolating network specific information flow from service specific information flow.

1. INTRODUCTION

Service Creation, when used with telecommunications technically comprises the processes of harnessing technological methods to facilitate the rapid prototyping, developing, testing, deploying and subsequent

maintaining of services. The field of Intelligent Networks places extreme demands on systems in terms of design, flexibility, scalability and performance, while inherently remaining a highly distributed model. It is thus only logical that such systems evolve to become heavily influenced by advances in distributed processing, especially CORBA.

Also one cannot build software-based telecommunications switching or control products without taking into account the quality and reliability of services the underlying protocols provide. The current trend in IN services is clearly evolving towards decoupling Service Creation and Service Execution environments from the underlying switch networks. A GSM Phase 2+ standard called CAMEL already defines the IN-GSM service interoperation, which allows mobile subscribers to access IN services [1]. Within the area of Intelligent Network Service Creation, there is always an implicit need to constantly reconcile many areas of service and protocol engineering. Although the service designer should not concern himself with signal-specific issues, protocols may change or evolve. Any framework used for service creation should thus be able to address this need to work at the protocol specific aspects as well, so that the same IN services can utilize several types of varying underlying protocol services.

However, at times standards-compliant implementations resulting from applying the multi-plane Intelligent Network specifications are cumbersome or not even possible, leading to many vendor-specific methods of service creation. This in turn directly warrants a need to design a simple, consistent and easy-to-use open framework or environment which, while still supporting rapid service development, can be used in a distributed environment in a standards-conformant manner to reap the advantage of load-processing over multiple machines.

Such an environment was successfully designed by building an experimental IN-specific framework which uses a few core components of an asynchronous event-based protocol implementation environment, OVOPS as a foundational basis. Enough parallels were seen in the areas of protocol implementation and IN Service Creation to be able to apply software practices and techniques learnt in the former to build a framework for the latter.

OVOPS though, only provides simple distribution chiefly through socket based communication techniques without any CORBA-based support for distributed computing. However, the OVOPS I/O Handler's principle of operation closely corresponds to the Reactor Pattern [2] as presented by Schmidt. This allowed the I/O handling of OVOPS to be extended to support CORBA invocations by introducing OVOPS specific device classes with functionalities appropriate to different ORBS. These can then handle

CORBA events external to the OVOPS system but trapped by the ORBs, as described in [3].

Thus the resulting framework is able to provide reliable native support for both the asynchronous and synchronous models of communication, the former for protocol specific transmission, the latter for making object calls for service execution while remaining CORBA compliant. This implies a reduced overhead for CORBA services such as the Event Service [4] and other services such as the Messaging and Notification Services [5], [6] that are being actively studied within the OMG.

2. THE FRAMEWORK ARCHITECTURE

The diagram in Figure 1 illustrates the framework which serves as an environment for both service creation and service execution. A Service Control platform provides a Service Logic Execution Environment (SLEE) within which Service Logic Processing Programs (SLPs) run to provide pertinent service processing. An SLP is a service application program invoked by the SLEE and is used to realize service processing under the control of the SLEE. The simultaneous invocation and execution of multiple SLPs are also managed by the SLEE. When an SLP is selected and invoked, it is referred to as a Service Logic Processing Program Instance (SLPI). In contrast to an SLP, a corresponding SLPI is a dynamic entity that actively controls the flow of service execution. This flow of service execution is broken down into a sequence of Service Independent Building Blocks (SIBs).

The CoreINAP Driver provides an implementation of the ETSI Core Intelligent Network Application Protocol (INAP) which is a Remote Operations Service Element (ROSE) user protocol. [7] The CoreINAP Driver connects to the underlying SS7 to communicate with other Service Points in the network. The Naming Service is a lightweight implementation of the CORBA Naming Service specification. The CoreXI provides a powerful generic request-and-response type interface to the database.

The CoreINAP Driver, Naming Service, and Database Driver components of the architecture are supplied by a third party vendor for the HP-UX 10.20 platform while the service specific factories, their associated call instance handling tasks as well as the SIB Inventory had been developed and execute on the Solaris 2.5 platform. Hence realizing the resulting framework lay in implementing the components which comprise the Service Logic Program (SLP) Factories, SLP Instances (SLPIs) and the SIB Inventory together with ensuring that they are designed to interact effectively via the IDL interfaces of the CoreINAP Driver, Naming Service,

and Database Driver components. These will constitute the Service Logic Execution Environment as well as a significant proportion of the Service Creation Environment (SCE) [8].

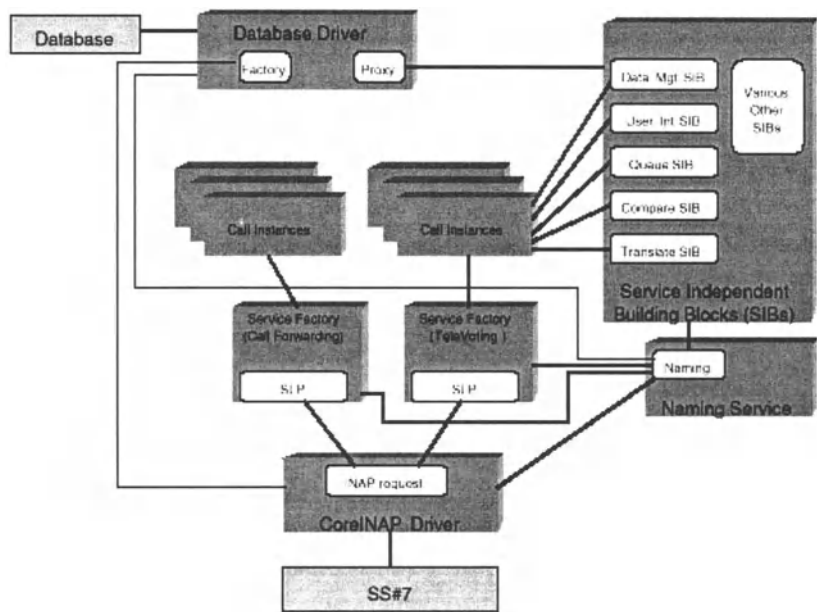


Figure 1. Framework Architecture

2.1 The SLP Factory

The design of the SLP Factory was aimed at supplying a service creator a generic class to derive his concrete service factories from. It furnishes an abstract interface for creating and destroying the various kinds of service logic instances. This provides consistency in the creation of new services and service factories. The interface also declares an abstract functionality of registration with the Naming Service which the instantiated service specific concrete factory can then override.

Apart from the above, it possesses a concrete interface together with operations for converting protocol-specific primitives (in this case INAP Operation primitives) into more generic information flows which can sufficiently supply the SLPIs as well as the SIBs with the information they need. Should the choice of the underlying protocol change, the propagation of the change would be greatly localized. The service designer needs only change the mapping element within the factory instead of having to redesign

all services from scratch, as shown in Figure 2. Table 1 shows how the mapping is done between the two.

Table 1. Mapping of INAP Operations to Information Flows

INAP Operation	Information Flow
Connect	Conn
ConnectToResource	ConnToRes
DisconnectForwardConnection	DisconnectForwardConnection
EstablishTemporaryConnection	EstablishTemporaryConnection
EventReportBCSM(oAbandon)	EndOfCall
EventReportBCSM(oAnswer)	CallAccept
EventReportBCSM(oCalledPartyBusy)	Busy
EventReportBCSM(oDisconnect)	EndOfCall
EventReportBCSM(oNoAnswer)	NoAnswer
EventReportBCSM(routeSelectFailure)	RouteSelectFailure
InitialDP	CallOrig
InitiateCallAttempt	InitiateCallAttempt
PlayAnnouncement	PlayAnnouncement
ReleaseCall	ReleaseCall
RequestReportBCSMEvent	NotifyBCSMEvents

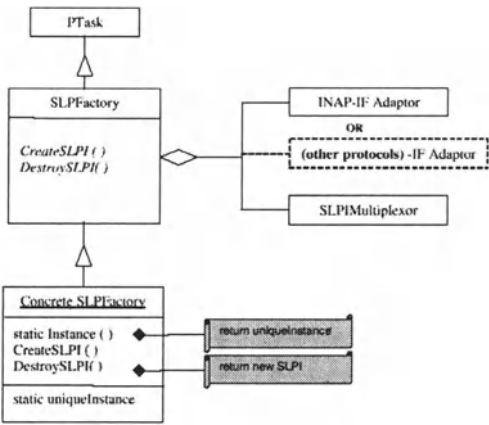


Figure 2. Creating a concrete SLP Factory

Lastly, the SLP Factory provides a concrete implementation of a multiplexing service to effectively handle the multiple instances of the services responding to the calls, one SLPI per call.

The implementation of the SLP Factory was achieved through a modified Abstract Factory Pattern [9] to take into account the INAP-specific and multiplexing nature of the resultant factory. Figure 2 illustrates the creation of a simple concrete Factory. The resulting concrete factories are usually

enforced with the Singleton pattern, since every application typically needs only one instance of a concrete Service Factory per service.

It can be easily seen that many of the INAP Operations have a natural one-to-one correspondence to their respective mapped Information Flow elements. However, some composite INAP Operations such as EventReportBCSM are decomposed into their individual contextual and semantic representations at the Information Flow level. The usage of an Information Flow ideology stems from the potentiality to map the entire dataflow at the same level in the IN Conceptual Model (INCM) as the Distributed Functional Plane (DFP). Information flows in the DFP are between two functional entities which either consist of a client request/server response pair or of a client request alone [10].

The SLP Factory was also designed to present a unified viewpoint of the SCE and the only gateway to the protocol engineer for each particular service; all calls which require a particular service would be channelled to the single concrete SLPFactory for that service, which will subsequently re-route the call to a dedicated SLPI to handle that call.

2.2 The SLP Instances

The SLPTask class serves as a base from which all SLPIs are derived from. The role of the SLPTask is three-fold; it firstly provides the necessary support at the framework level needed by each call instance for proper service execution. It also acts as the Global Service Logic (GSL) which serves as a storage area for data, or glue, essential for the proper interaction and subsequent service realization by the SIBs, and lastly it partially models the BCP to provide Points of Initiation (POIs) and Points of Return (PORs).

SLPTasks are created and destroyed by the SLPFactory to which they belong to. Call instance routing is done from the SLPFactory to the SLPTask via the SLPTask's *getID ()* function call, which returns a unique identifier with which the SLPFactory distinguishes each call instance.

2.3 The SIB Inventory

The SIB Inventory provides a subset of fully implemented SIBs which can be easily used as building blocks for a service. All SIBs within the inventory are implemented as objects which register themselves with the Naming Service upon startup. The SIB Inventory presently supports only synchronous communication techniques. The *start()* function call is responsible for launching the logic of the SIB. The test framework has implemented five of the fourteen SIBs defined: The Compare, Translate,

Service Data Management SIB, the User Interface SIB and the Basic Call Process (BCP) [11].

One of the driving design issues of building a SIB Inventory is that the instantiation of the SIBs should be as minimal as possible to avoid a proliferation of an extremely large number of SIB instances for a moderate to complex service design. At any one point in time, service implementations, which would utilize several different kinds of SIBs, should ideally use only one of each kind of SIB. Thus the SIBs have been designed following the principles of design similar to the Flyweight Pattern [9].

However, the usage of CORBA means that the Flyweight Factory is no longer necessary, as it can be replaced by the ORB handling all operation invocations on the server SIB objects without any undue interference and with little loss in performance. All objects within the same server process on a given machine can be managed by the same process on that machine. Some ORBs such as Orbix also support an activation mode in which only one instance of a server object is launched to handle multiple client requests. In addition, if a server is registered in the shared mode, it can be launched manually, prior to any invocations on its objects [12].

The principles of how this model interacts and interworks with an SLPI (meaning an SLP instance of a concrete SLPTask-derived class) is described in the next subsection.

2.4 Interworking between the SLP Instances and the SIB Inventory

Since each SLPI is an event-driven task, it possesses state information and depending on the current state and the type of Information Flow (IF) it receives from its concrete service factory, it launches a predetermined chain of SIBs to perform the necessary series of actions. The SLPI thus forms the “glue” which holds all the necessary data and parameters needed by the SIBs, since the SIBs themselves are independent blocks which have no notion of service. Therefore at the GFP, the SLPI would also perform the functions of the GSL. At the Service Plane (SP) level, the entire service should appear seamlessly integrated, so clear IDL interfaces are needed for both the SIB implementations and the SLPI which need to be known at compile time for data to be interchanged efficiently to realize such IN services.

In order to interwork harmoniously, following the flyweight pattern, the SLPIs would behave as clients while each SIB would be represented as a ConcreteFlyweight, each having a similar interface to the others, differing only in terms of the names used to register with the Naming Service. However, flyweights cannot assume the context in which they are operating.

As the SIBs will be unable to store any extrinsic information, all data uniquely pertaining to the entire duration of a call, or to a temporally specific moment within the call, need to be held by the client, which in this case refers to the SLPI. Such data would usually involve the CID and SSD parameters for use by all the SIBs. The next subsection will describe how the SSD is stored within the SLPI.

2.4.1 Storage of the SSD in the SLPI

The variable information which the SLPI needs to store varies not only with the state of the SLPI, but also with the type of input the SLPI receives. Thus if an SLP Instance has two states, Idle and Active, and is able to receive two kinds of Information Flows, Call_Begin and Call_End, a total of four different SIB chain combinations need to be defined.

The State Pattern [9] when modified, provides an easy behavioural technique for managing this complexity. The key idea is to introduce an abstract class called SIBChain which represents the state*input combinations of the SLP Instance. The SIBChain class declares an interface common to all the classes that represent the different operational permutations. Subclasses of the SIBChain implement the specific combinatorial behaviour. For example, the classes Idle_Call_Begin and Active_Call_Begin specify and implement the behaviour pertinent to that particular SIB chain. Figure 3 illustrates how this was accomplished.

The classes can also be shared between SLPIs using the Singleton Pattern. However, the introduction of the SIBChain by itself does not solve the problem of SSD storage. Rather, it provides a logical breakdown of the different numbers of SIB chains necessary for the service execution. Also, by virtue of its semantics, the SSD will change each time the SIB is reused or changed, and hence, a more dynamic form of representation is necessary for such a storage. This therefore leads to the usage of proxy objects for the SIB flyweights, the SIBNodes.

By using node pointers, inheritance as well as SIB-specific internal data representation to store the SSD of the various SIBs they represent, the SIBNodes present a common interface to the SLPI to access the SSD needed by the next SIB in the chain when it is requested, and systematically present an ordered and organized way to specify the service in terms of its SSD. Five of these proxy SIBNodes are implemented, corresponding to the five SIBs implemented within the SIB Inventory. The following illustrations show the object diagram depicting this.

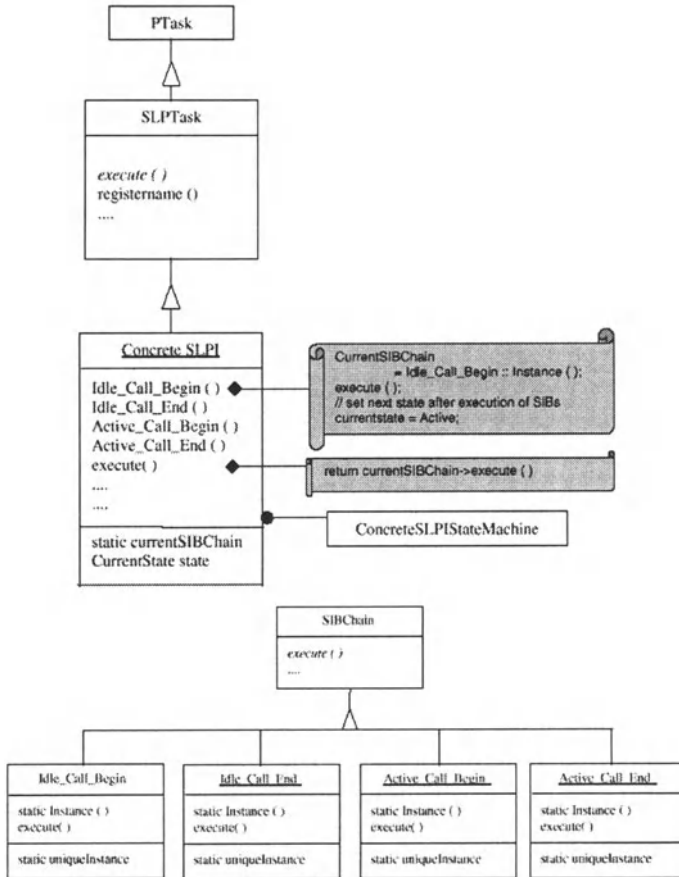


Figure 3. Usage of the modified State Pattern

Each SIBChain derived class is responsible for the construction of a tree-like branching chain using these SIBNode derived classes upon creation, and then initializing them to contain the relevant SSD information used for the proper execution by the SIBs upon reaching that level. However, it remains the SLPTask's responsibility for maintaining the current status of processing within that tree. Upon completion of processing, the current SIB will request the SLPTask for information of the next SIB to which control should be passed through the SLPTask's *getnextsib ()* function call. This call will return the next SIB and its SSD in the chain, depending of course on the results of execution by the current SIB. Thus, using such a technique, a service designer is able to organize session specific data in a manner logical and organized enough for the SLPTask to control, manage and interpret while being able to interwork with the necessary SIBs.

2.4.2 Storage of the CID in the SLPI

The storage and retrieval of CID per SLPI by the SIBs proceed in a different manner as compared to the complexity of managing the SSD. Far from being context-specific, the information stored as CID does not change for the duration of the call instance.

Thus the best technique possible to store the CID would be within the SLPI itself, with the SLPTask providing an interface with which the SIB can retrieve or manipulate the CID using direct object calls. Currently the framework implementation supports, implements and returns such pertinent function calls made by the SIBs in the Inventory to the SLPTask requesting call instance specific data such as the called party number as well as the calling party number. Other kinds of CID operations are left as virtual operations for implementation at the service designer’s liberty.

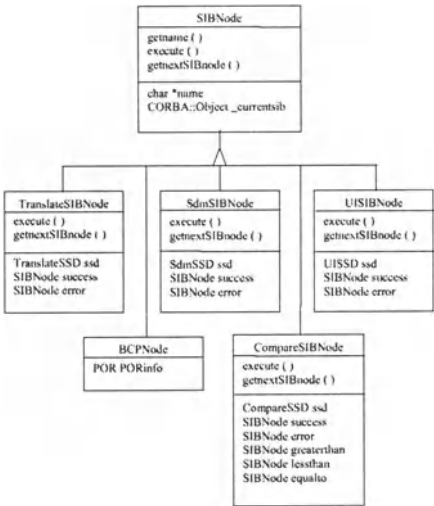


Figure 4. SIBNode class derivation

3. AN EXAMPLE SERVICE CREATION SCENARIO

The Number Translation Service (NTS) is a simple service where numbers beginning with a certain sequence trigger off IN service processing logic. The service basically centers around 4 different phone numbers: numbers A, B, C and D. As an example, if someone telephones number A, the call will be rerouted to either B, C or D, depending whether it is during working hours, after working hours or during a holiday. If the rerouted number is unreachable, busy or unanswered, the service logic would then

proceed to connect the caller to a default number. As such, the NTS service logic needs to make queries to the database to obtain the correct number to connect to in the telephone network.

The functionality of the SIB chain that is to be launched upon reception of the *CallOrig* IF in the *Closed* state is shown in Figure 5. Three SIBs from the SIB Inventory were used to realize this particular SIB Chain, with the *CompareSIB* and the *TranslateSIB* being used multiple times.

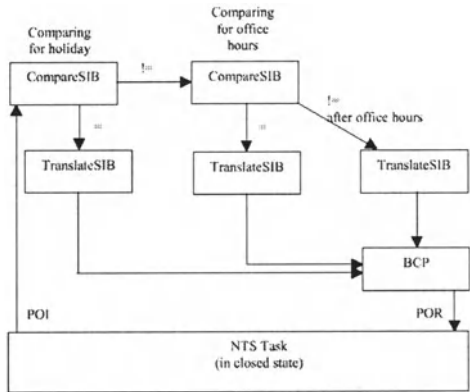


Figure 5. Chained SIBs upon receiving the *CallOrig* IF

4. DESIGN DECISIONS

The framework presents a model in which complexity is hidden via base classes and a single unified interface is presented at key access points to its components. As a consequence, the entire SLEE is viewed at from only one viewpoint from the underlying application specific protocols such as INAP. Such principles of component oriented programming have proven extremely useful in managing and reducing the problem domain into clear and exclusive subdomains which could then be tackled easily.

Most, if not all, of the Design Patterns used in this work would have been of little use if they had been used "as is". In some cases, modifications to harness the distributed nature of the environment were highly necessary. However, once altered and adapted to work with such distribution, they seemed to support and work seamlessly with the rest of the design of the system.

The modified State Pattern distributes behaviour for the different combinations across several subclasses. This increases the number of classes and is less compact than a single class. However, such a distribution is

actually good, because given a large permutation, the only other way to solve it in a single class would be to necessitate large conditional statements. The usage of the State Pattern was preferred over the more traditional table-based alternative due to the fact that this pattern allows the localization of state-specific behaviour into one object. An added bonus is that since state objects can be shared if they have no instance variables, they essentially become flyweights with no intrinsic behaviour.

Often, such State Patterns are combined with the Singleton. However, the drawback here of employing the Singleton Pattern with the State Pattern is the one basic premise that class instances created by enforcing the design technique introduced in that particular pattern are expected to be persistent throughout the application. This might not be appropriate for the classes adhering to the modified State Pattern, as they need only exist as long as concrete SLPs implementing the service and representing Call Instances exist. Otherwise they should be deleted. Many algorithms can be proposed to implement such a scheme. Perhaps the most straightforward would be a modified Singleton with a relatively simple reference-counting garbage collection technique. This technique was employed successfully in building the example Number Translation Service (NTS).

The concepts introduced by the utilization of the Information Flows and the BCP were applied liberally. It allows the framework to be robust enough to withstand variations in underlying protocol services as well as being capable of extending towards rudimentary “Plug-and-Play” type support at the protocol level. Information Flows furnish a more natural way of modeling service logic input than INAP service primitives provide.

Modeling the BCP could not be easily achieved by treating the SLPTask as a BCP by itself. Thus the BCP was modeled in a fashion similar to other SIBs, the difference being that the BCP defines the PORs by which the SIB chain ends. The PORs were mapped into function calls in the SLPTask, thereby bringing control gracefully back from the SIB chain. From this perspective, POIs could be easily mapped into events, but PORs were much better mapped as function calls. Keeping the SIBs purely synchronous resulted in a set of objects whose co-operation was easier to manage.

The design of the SIB and abstract SLP classes are done in such a way that interworking between the SIB Inventory and the concrete SLPs can be achieved without much difficulty. However, close co-operation is needed at least at the framework level to be able to successfully achieve mapping on both the GFP and the SP in the INCM. Although in principle many services other than the NTS can be implemented with these SIBs such as those described in [13], the application programming interface for the SIBs was not specified in IN CS-1. One other issue which also lay in mapping the different planes of the INCM was the fact that the entire specified model has

been mostly based on a functional and modular data flow approach. This made the task of denoting the same relationship with objects that express such associations as clearly, open to some interpretation.

However, although SIBs were specified as abstract entities which must be decomposed into a set of interacting FEs and FEAs at the DFP, the experience gained from this work shows that using an object oriented model, the realization of SIBs as distinct and concrete implementations which provide unified and stable interfaces is possible as the usage of CORBA technology has led to a credible usage of these building blocks.

Subsequently, the prototyping of simple IN services can be achieved extremely rapidly as compared to more traditional SCEs at which reuse may lie either at the network or application resource level. It may not be unrealistic to note that the development time schedule for the implementation of fully working IN services may be reduced orders of magnitudes using a framework built with a similar design objective as this.

5. CONCLUSIONS

The presented framework is compliant with the core CORBA 2.2 specifications [14] as well as the INCM. Conformance to the TINA model can be observed within the confined context of CS-1. A one-to-one mapping between the INCM and the TINA conceptual framework is not entirely possible, but several parallels can be drawn from both models which allow a partial mapping, such as the mapping of the SIBs to TINA service objects/logic, as well as the mapping of some TINA computational objects at both the DFP and the GFP [15]. In the long run, it must be borne in mind that the IN Capability Sets are still evolving.

The OMG working group for IN/CORBA Interworking has proposed two orthogonal scenarios of using CORBA in IN Signaling. The first may be identified as a gateway which allows CORBA-based IN application entities to interwork with those that communicate using the INAP, while the second would involve using the existing Signalling System No.7 (SS7) infrastructure as a transport network for GIOP messages[16]. Of the two, the first type of interworking, which is what this framework has attempted, is expected to be the more prevalent of the two as a principal and immediate solution.

In conclusion, it is imperative to mention that interworking between CORBA and Intelligent Networks Systems is still under active study and the work presented in this paper uses, discusses and attempts to find a working solution based on these concepts and ideas.

REFERENCES

- [1] Intelitel Communications: White Paper on Network Intelligence Middleware, 1999
- [2] D. C. Schmidt: Reactor: An Object Behavioural Pattern for Concurrent Event Demultiplexing and Event Handler Dispatching. Pattern Languages of Program Design (J.O. Coplien and D. C. Schmidt, eds.), Reading, MA: Addison-Wesley, 1995.
- [3] B. Silverajan, J. Harju: OVOPS – Enhancing an Event Based OO Framework for Distributed Programming, Proc. TOOLS USA '99, Santa Barbara, CA August 1 - 5, 1999.
- [4] OMG : CORBAservices, Common Object Services Specification, December 1998
- [5] OMG : Messaging Service Document orbos/98-03-12
- [6] OMG : Notification Service Document telecom/98-01-01
- [7] ETSI: Recommendations ETS 300 374-1, Intelligent Network Capability Set 1 (CS1) Core Intelligent Network Application Protocol (INAP) Part 1: Protocol Specification, ETSI 1994.
- [8] B. Silverajan: A CORBA-Enhanced Implementation Environment for Distributed IN Service Creation, MSc Thesis, 1998.
- [9] E. Gamma, R. Helm, R. Johnson, J. Vlissides: Design Patterns, Elements of Reusable Object-Oriented Software. Addison-Wesley 1995.
- [10] ITU-T Recommendation Q.1204 : Intelligent Network Distributed Functional Plane Architecture, 1993
- [11] ITU-T Recommendation Q.1213 : Global Functional Plane For Intelligent Network CS-1, 1993
- [12] IONA Technologies : Orbix 2.2 Reference Guide, March 1997
- [13] ITU-T Recommendation Q.1211 : Introduction to Intelligent Network Capability Set 1, 1993
- [14] OMG: The Common Object Request Broker: Architecture and Specification. CORBA V2.2, February 1998
- [15] Demounem, L.: Intelligent Networks and TINA Service Models – Present and Future, Proceedings of the IFIP/ICCC International Conference on Information Network and Data Communication (INDC '96), Trondheim, 1996
- [16] OMG: IN/CORBA Interworking, document telecom/98-10-03

FROM SIBS TO DISTRIBUTED OBJECTS: A TRANSFORMATION APPROACH FOR SERVICE CREATION¹

Elie Najm, Frank Olsen, Sylvie Vignes

École Nationale Supérieure des Télécommunications

Elie.Najm@enst.fr, olsen@acm.org

Abstract This paper describes how to apply correctness-preserving transformations to the problem of service creation using the IN CS-1 SIB (Service Independent Block) concept. We show how SIBs can be represented by a UML class diagram augmented with constructs from an action language and then automatically translated into a Java like language. We also show how this transformation can be implemented using a UML-based CASE-tool.

1 INTRODUCTION

This paper presents our work on service creation for telecommunications networks. We position ourselves in a world where the predominant network architecture is the Intelligent Network (IN). However, the IN has many well-known limitations. In this paper we consider a major one: The use of a completely algorithmic approach to service creation and reuse of “components”. This problem has been acknowledged by most of the actors in the telecommunications business and has led to interest in new and improved architectures based on objects and distributed computing technologies. The major example of these new architectures is TINA.

The starting point for this paper is our previous work on correctness-preserving transformations for open object-based distributed systems [1]. There we defined an automatic transformation between two languages: a *source language* (*SL*) and a *target language* (*TL*). *SL* is based on the assumption of operation on a global state while *TL* is a representative of distributed object-based languages.

Our main contribution in the present paper is to apply our transformation for IN Service Features (SIB chains). In other words, we are interested in

¹This work has been partially supported by RNRT project PILOTE

a top-down approach aided by automatic transformations. Since the result of the transformation is an object-based language, our work also provides a possible evolution path from IN-style service creation towards open object-based distributed systems as exemplified by CORBA or TINA.

1.1 Outline of the paper

In section 2 we give an overview of the IN architecture and outline the current process used for creating IN services. Next we introduce the main results from our previous paper mentioned above that are used as a basis for the current paper in section 3. Then comes the central part of the paper where we provide a mapping of SIBs (and Service Feature) into distributed objects in a Java-like language. Finally, we conclude in section 5 with a brief summary of this contribution, a comparison with existing work, and an outlook for further work.

2 IN SERVICE CREATION

In this section we give an overview of the Intelligent Network architecture and with emphasis on the service creation aspect.

2.1 The Intelligent Network architecture

Work on the IN architecture (ITU-T Recommendation Series Q.12xy [3]) started in 1988 with the aim of making service creation more rapid than for the existing telecommunications network. Before the IN, all the intelligence was in the switches and service creation was performed by the switch (hardware) providers (on demand from the telecommunications operators). The main solution proposed by the IN was to separate the service logic from the call and connection control. The latter were still performed in the switches, but the former was separated out into a new piece of equipment called a Service Control Point (SCP). The different functions of the network are performed by Functional Entities (FEs); the main FEs defined by the IN are the CCF (Call Control Function) which performs call and connection control, the SSF (Service Switching Function), and the SCF (Service Control Function) which contains the service logic. Typically, the SCF is located in the SCP and the CCF and SSF in the switches (also called Service Switching Points (SSP)). The role of the SSF is to hand over control from the SSP to the SCP at specific points in a call where service logic can be invoked. A typical example is that a user dials a number which the SSF hands over to the SCF: the SCF can now invoke a feature like Originating Call Screening (OCS) which verifies that the dialed number is not on a list of screened numbers.

Acknowledging that the telecommunications world is in constant evolution with new technologies constantly appearing, the IN is defined in phases called

Capability Sets (CS). As of mid 1999, two CSs have been released and a third one is due at the end of 1999. Simultaneously, work has already started on a fourth capability set. In this paper we only take into account IN CS-1 (Q.121x) [5].

2.2 Intelligent Network Conceptual Model

The Intelligent Network Conceptual Model (INCM) provides a conceptual framework for the provisioning of telecommunications services and service features. It defines four different planes: The Service Plane (SP), the Global Functional Plane (GFP), the Distributed Functional Plane (DFP), and the Physical Plane (PP). These planes define different levels of abstraction of the telecommunications network and its services. In this paper we will not discuss the Service Plane, since it is not defined for IN CS-1; suffice to say that the SP deals with Services and its constituent Service Features.

In relation to service creation the main idea behind the INCM is to describe Services and Service Features at different levels of abstraction thus enabling people with different skills and detail of knowledge to be actors in the service creation process. The skills range from business-related skills at the SP to detailed knowledge of the network and its architecture at the PP where new network functionality may need to be implemented to create a new service.

Global Functional Plane. The GFP provides a set of Service Independent Building Blocks (SIB) which are used to create Service Features. A SIB is a procedural building block that can be reused in several Service Features.

A Service Feature is a flow graph of SIBs (also called a SIB chain) with the outputs from one SIB connected to the input of another. The starting point for a service feature is a POI (Point Of Invocation) which corresponds to a specific point in the BCM (Basic Call Model). The BCM is a state machine representation of a telephone call. A service feature has one or more PORs (Points Of Return) which return control to specific points in the BCM.

Distributed Functional Plane. In the Distributed Functional Plane (DFP), the SIBs at the GFP are decomposed into Functional Entity Actions (FEA). Each FEA belongs to a single Functional Entity (FE). A single SIB can be distributed over several FEs: In this case there will be several FEAs for the SIB and they will communicate using Information Flows (IF). The parameters passed with the IFs are called Information Elements (IE).

Physical Plane. The basis for the Physical Plane (PP) of the IN architecture is the *Signalling System no. 7* (SS7) which is the protocol stack (up to the Application Layer of the OSI seven-layer model) that supports the *Intelligent*

Network Application Protocol (INAP). INAP uses the services provided by *Transaction Capabilities Application Part* (TCAP).

At the center of TCAP (or more precisely of the *Component Sub-layer*) lies the *ISO Remote Operations Service* (ROS) that provides support for distributed objects. ROS describes interactions between distributed objects using the *Abstract Syntax Notation One* (ASN.1) *information object class* (or ASN.1 macros in older versions of the ITU-T recommendations). The *Transaction Sub-layer* is then used to provide a “skinny” end-to-end connection for transporting operations between Functional Entities.

3 TRANSFORMATIONS FROM ACTIONS TO DISTRIBUTED OBJECTS

As mentioned in the introduction, the present paper extends previous work [1]. In this section we give a very brief overview of the previous paper in order to make this contribution more or less self-contained¹.

Thus, in [1] we presented a correctness-preserving transformation from a source language (*SL*) - capturing a centralised view of services - to a target language (*TL*) where distribution is taken into account. Here we provide a reminder of the two languages through an example.

The source language for our transformation is a simple imperative class-based language with actions operating on a global state. The following simple bank account example shows the main features of the language:

```
Account = class owner: string;
          balance: integer;
          endclass

(* the following action transfers a sum equal to amount from the *)
(* provider account to the recipient account *)

transfer(provider:Account; recipient:Account; amount:positive) =
action
case provider.balance < amount: return NACK;
case provider.balance >= amount:
    provider.balance := provider.balance - amount;
    recipient.balance := recipient.balance + amount; return ACK;
endaction
```

¹Please refer to the previous paper if more detail is needed: Notably there is an emphasis on the formal aspects of the transformation, including the operational semantics of both the source and target languages.

The main features of *SL* is that there are two kinds of classes: *data classes* containing only attributes (represented by the *Account* class in the example, and *action classes* providing the behaviour (represented by the *transfer* action). The most prominent feature of the transformation itself is that assignments are mapped into remote method invocations thereby introducing a distribution aspect. The target language is a distributed object-based language which can be compared to Corba compliant languages or to Java enhanced with RMI. To understand the target language and the transformation we now return to the example given in *SL* above after having run it through the transformation. This gives the following program in *TL*:

```

Act_Account(owner, balance) =
  [get_owner(ret) ->
    ret.value(owner);
    become Act_Account(owner, balance),
  get_balance(ret) ->
    ret.value(balance);
    become Act_Account(owner, balance),
  set_owner(new_owner) ->
    become Act_Account(new_owner, balance),
  set_balance(new_balance) ->
    become Act_Account(owner, new_balance)
  ]
Act_Transfer() =
  [transfer(provider, recipient, amount, return) ->
    provider.get_balance(self);
    [value(p_balance) ->
      if p_balance < amount then return.value(NACK)
      else recipient.get_balance(self);
      [value(r_balance) ->
        provider.set_balance(p_balance-amount);
        recipient.set_balance(r_balance+amount);
        return.value(ACK)
      ]
    ]
  ]
]

```

The target language considers active objects (called *actors* - thus the prefix *Act_*). The data class has become an object *Act_Account* where the attributes have been transformed into accessor functions prefixed by *get_* and *set_*. The *action class* has been transformed into another object *Act_Transfer*. We will

not go into any more detail about *TL*, since the language used in this paper, although inspired from *TL*, has a syntax closer to Java.

4 TRANSFORMING SIBS INTO DISTRIBUTED OBJECTS

This section forms the central part the paper. It presents a translation from IN Service Features and SIBs to distributed objects.

4.1 Translating individual SIBs

We have already given an overview of SIBs in section 2.2 where we discussed the GFP of the INCM. Although SIBs appear on several of the INCM planes with differing levels of abstraction we concentrate on the GFP since our contribution is on a *top-down approach* to service creation where distribution aspects is provided by the transformation. So, let us now remind ourselves of the SIB model as presented in the ITU-T CS-1 Recommendations [5] (Q.1213)².

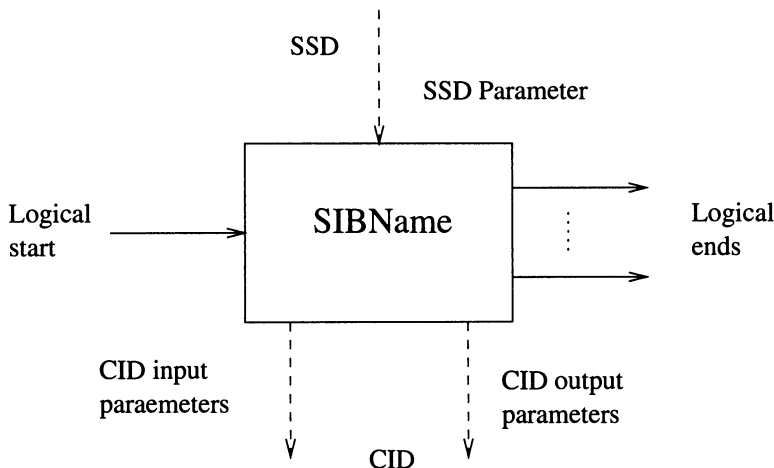


Figure 1 SIB model for the GFP.

Figure 1 shows the graphical representation of CS-1 SIBs. The *CID* (Call Instance Data) parameters describe the dynamic input and output of values to and from the SIB at run-time. To make SIBs usable in several contexts (i.e., in several Service Features) the *SSD* (Service Support Data) are used to specialise a SIB for use in a specific Service Feature.

²Later we may extend our work to take into account CS-2 and later capability sets.

So far we have given the impression that only the GFP is used for modelling SIBs. However, to implement SIBs we also need information from the DFP (see section 2.2). The reason is that the SIB concept has been conceived by studying the existing telecommunications network; i.e., it has been created in a bottom-up fashion. It thus is dependent on the underlying network architecture provided by the DFP and PP, notably in that it is implemented using INAP operations between Functional Entities (see section 2.2). Note that so far we have only taken into account the SCF, SDF, and SRF. Also we do not model the BCSM in any way.

Although SIBs could be modeled directly in *SL*, we prefer instead to capture them using UML. As we explain before, UML allows us to easily implement the transformation using the *Objectteering* UML CASE-tool. However, the UML is augmented with constructs from *SL* to provide an *implementation* for the SIB functionality using the *action classes* described in an earlier section.

The capture of SIBs is into two classes: One action class containing the SSD together with a method that implements the SIB from the SCF point of view, and another data class that contains the CID. A relation between these two classes links the static part of the SIB with the dynamic (CID) part. Thus a SIB is represented by a UML class diagram as shown in figure 2.

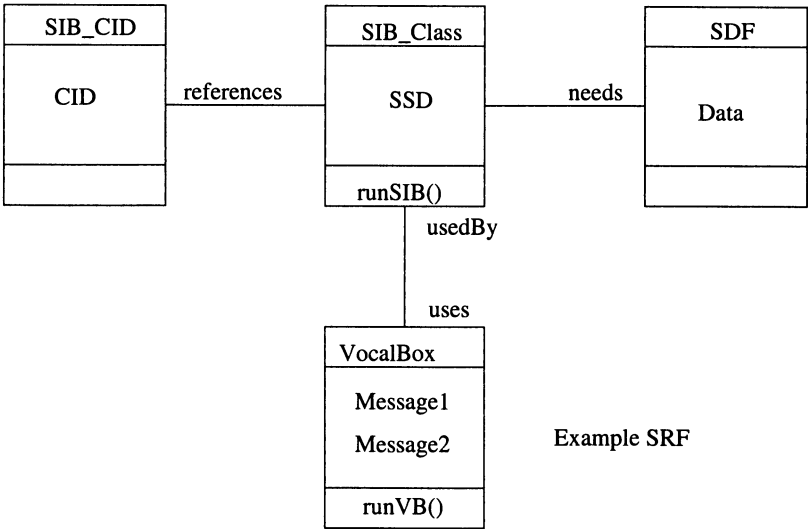


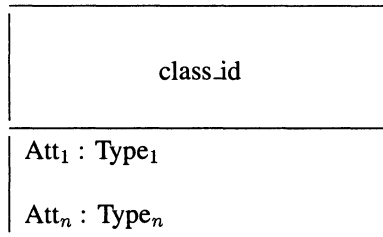
Figure 2 SIB mapped into a UML class diagram.

Figure 2 only shows the pure UML part of our mapping: For the implementation of the SIB functionality (the runSIB method in the SIB_Class) we

use *SL* as comments attached to the *SIB_Class*. We model Service Features as a UML class with a single method called *main* whose implementation is given in *SL*. Since Service Features are modelled as SIB chains, there is a need for a SIB to return a value corresponding to its *logical end*. We do this by adding an attribute called *returnSIB* (of an enumerated type) to the *SIB_Class*.

4.2 Transformation to distributed objects

So far we have shown how SIBs and Service Features are represented in UML class diagrams annotated with *SL* constructs in comments. In this section we show how this new source representation (UML + *SL*) is transformed into a variant of *TL* based on the transformation from [1]. The transformation has been slightly modified to add a *private* section to the transformed class: All the attributes of the UML class are placed in the private section to preserve encapsulation. The accessor methods for these attributes are retained from the original transformation. We also add a constructor with no arguments (for classes without any attributes), as well as a constructor with all attributes as parameters for classes containing data. Thus, the following data class containing attributes:



is transformed into the class below:

```
class class_id
{
  //-----
  //ATTRIBUTES
  //-----
  private T[Type_i] att_i;

  //-----
  //CONSTRUCTORS
  //-----
  public class_id () {};
  public class_id ( ..., T[Type_i] att_i, ... )
  {
```

```

    ...
    this.att_i = att_i;
    ...
}

//-----
//ACCESSORS
//-----
public T[Type_i] get_att_i ()
{
    return att_i;
}
public set_att_i ( T[Type_i] att_i )
{
    this.att_i = att_i;
}
}

```

It is important to note that the notation $T[Type\_i]$ does *not* represent a construct in the target language: It is the notation we use to show the mapping of types. The same is true for all the transformation rules. It is due to the fact that the transformation is done in stages.

For the remainder of the transformation rules we refer to the appendix. Instead we now go on to show a practical example of what we can do.

4.3 An example: TeleVote

We now turn to an example to give a more intuitive impression of what we can do with our transformations. We use the *TeleVote* Service Feature given in [6] (see the conclusion for a comparison with this work).

Definition. After dialing the service access number for TeleVote the voting user receives a vocal message telling him/her to choose between two or more possible choices, each corresponding to a number on the telephone handset. Finally, a new message confirms that the choice has been taken into account.

GFP description. Figure 3 shows how TeleVote can be modelled on the GFP of IN CS-1.

User Interaction SIB. Figure 4 shows the UML model for the SIB User-Interaction according to our previous description in section 4.1. We do not show the other two SIBs involved in the TeleVote Service Feature since they are simpler than the UserInteraction SIB (they are purely algorithmic and do not communicate with any other Functional Entity). For the original

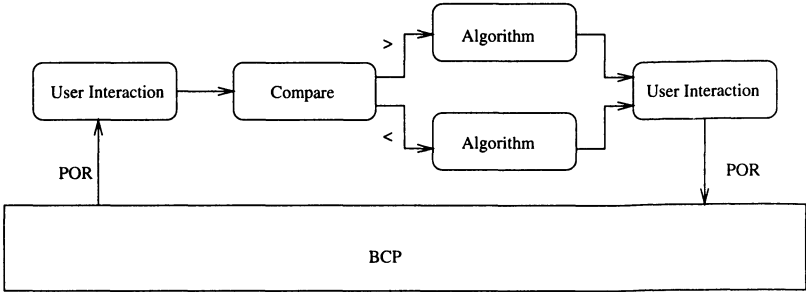


Figure 3 GFP model of TeleVote Service Feature.

description of the SIB we refer to the relevant ITU-T recommendation Q.1213 (the GFP) and Q.1214 (the DFP).

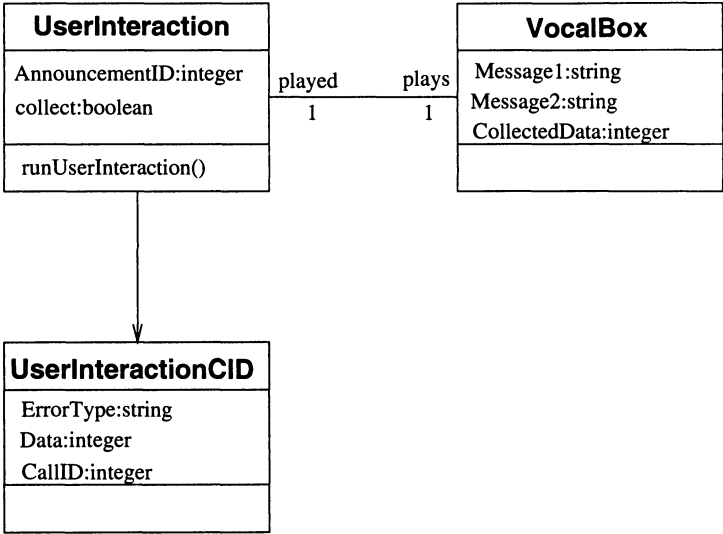


Figure 4 UML class diagram for User Interaction SIB.

The most interesting part of this diagram is the `VocalBox` which is part of the SRF interface. Note that due to limitations in the datatypes provided by *SL*, we do not model all the SSD of the SIB, since we do not have types to handle for instance dates and times used in the INAP operations that implement the communications part of this SIB.

4.4 Implementing the transformation

The translation has been implemented using a UML-tool called *Objectteering*³. The tool, in its current version, provides a language (called *H*⁴) that allows UML-models to be easily mapped into an implementation language.

As shown above, SIBs and Service Features are modelled in UML augmented with structures from the *SL* language. In fact, *SL* constructs appear as comments attached to the UML classes representing SIBs in UML class diagrams. These comments are taken into account by *H* to allow our previously defined transformation to be used to map SIB classes into *TL*. A trial implementation was conducted by a student project at ENST [4].

5 CONCLUSION

We have shown one possible way to evolve from the world of IN service creation to the one of open object-based distributed computing. This was done by mapping IN CS-1 Service Features and SIBs into a language resembling Java. The transformation is based on previous work on correctness-preserving transformations [1].

The work most closely related to ours is that of Nasreddine et al. [6]. They also map SIBs into classes, but instead of making the SSD a class attribute they provide it as a parameter to each call of the method implementing the SIB. Also, at least in the paper, they do not show how the SIBs are implemented. The biggest advantage of our work is that we provide a transformation from Service Features and SIBs to a language based on distributed objects that have been implemented. Unlike us, they have implemented their work using CORBA (although without taking into account the existing SS7-based network). Finally, a very notable feature of their work is that they aim for methodology for evolving existing algorithmic-based service creation towards object-oriented service creation, taking into account the various actors (with differing skills) that are involved in the service creation process. Instead, we have concentrated on a key issue: how to automate the passage from distribution transparent service specifications to distribution aware implementations.

Clearly, the work presented here is still in its early stages. Our current research focuses on comparing and integrating it with other research efforts on the evolution of the IN architecture. Notably, we want to make it possible for our SIBs to communicate with existing SS7-based Functional Entities. It is important to preserve the existing investment in SSPs; thus, we need a way for call events in the SSPs to trigger Service Features located in a CORBA

³Objectteering is a trademark from Softeam.

⁴The name of this language will soon be changed into *J* because it is moving towards something close to Java.

environment. There are many open questions in this exiting area of research. Both EURESCOM, the OMG and TINA-C have effected interesting work in this area, either through projects (like EURESCOM P508 and P847-GI) or RFIs and RFPs by OMG and TINA-C. Clearly, a pure CORBA environment is not currently suited to the real-time, high-reliability demands of telephony services, although recent extensions to CORBA will alleviate these problems: In particular, *CORBA 3* supports *Asynchronous Messaging* (all INAP operations are asynchronous), *Real-Time*, *Components*, and *Firewalls* (to allow callbacks through firewalls).

Appendix

Transformation rules from SIBs to distributed objects

Relations between classes are transformed as follows (showing only the mapping for one of the two classes in the relation).

```
class Class1
{
    //-----
    //ATTRIBUTES
    //-----
    private Class2 role2Class2;

    //-----
    //ACCESSORS
    //-----
    public get_role2Class2 ()
    {
        return role2Class2;
    }
    public set_role2Class2 ( Class2 role2Class2 )
    {
        att_i = new_value;
    }
}
```

Note that we use the names of roles played by each class in the relation together with the class name to construct a unique name for the reference to the other class.

Finally, the transformation of actions given below closely follows the one in [1]. (We do not give an explanation for the cases where the transformation is straightforward.)

Action header.

```

T [ Action ( ..., Parameter_i : Type_i ,...) : Type ]
->
public T [ Type ]
    Action ( ... , T [ Type_i ] Parameter_i , ...) ]

```

Unlike our original transformation we do not create a new thread to handle message execution. The reason is that Service Features are based on the sequential execution of SIBs. Synchronising the resulting system if threads were used would be very complex indeed.

Action body.

```

T [ Declaration    -> T [ Declaration ]
    beginaction      T [ Statement ]
    Statements
    endaction ]

```

The transformation is direct.

Variable declarations.

```

T [ Decl1Decl2 ]  -> T [ Decl1 ]
                  T [ Decl2 ]

T[ var : Type ]   -> T [ Type ] var;

```

The transformation is direct.

Statements. For statements we consider the cases separately.

```

T [ var:= Exp; ]      -> var = T [ Exp ];
T [ var.att := Exp; ] -> var.set_att( T [ Exp ] );
T [ if Exp then      -> if T [ Exp ]
    S1                {
    else S2            T [ S1 ]
    endif; ]          }
                     else
                     {
                     T [ S2 ]
                     }

T [ return Exp; ]     -> returnClassName = T [ Exp ];
T [ S1S2 ]            -> T [ S1 ] T [ S2 ]

```

We note that for assignments to an attribute we must use the generated accessor methods.

In the case of `FunctionCall` we get the following transformation.

```
T [ Object.Method ( ..., Exp, ...); ]
-> Object.Method ( ..., T [ Exp ], ... ) ;
T [ print(..., Exp, ...); ]
-> System.out.println ( ... + T [ Exp], ... );
T [ var := read() ]
->
var = Integer.parseInt((new java.io.BufferedReader
(new java.io.InputStreamReader(System.in)).readLine()));
```

Expressions.

```
T[c]          -> c;
T[var]        -> var;
T[var.att]    -> var.get_att;
T[nil]        -> null;

T [ new class_id ( ..., Exp_simple, ... ) ]
-> new class_id( ..., T [ Exp_simple ], ... );
T [ operation_id ( ..., Exp_simple, ... ) ]
-> operation_id( ..., T [ Exp_simple ], ... );
T [ action_id ( ..., Exp_simple, ... ) ]
-> action_id( ..., T [ Exp_simple ], ... );
```

Types. Class names are conserved.

`T[class_id]` -> `class_id`;

For the base types we obtain the following mapping.

<i>SL</i>	<i>TL</i>
integer	long
real	double
boolean	boolean
string	String

References

- [1] Cinzia Bernardeschi, Joubine Dustzadeh, Alessandro Fantechi, Elie Najm, Abdelkrim Nimour, and Frank Olsen. Consistent semantics and correct

transformations for the ODP information and computational models. In *Proceedings of Second IFIP conference on Formal Methods for Open Object-based Distributed Systems - FMOODS'97*. Canterbury UK, Chapman & Hall, July 1997.

- [2] I. G. Dufour, editor. *Network Intelligence*, volume 10 of *BT Telecommunications Series*. Chapman & Hall, first edition, 1997.
- [3] ITU-T. *Intelligent Network: Q.1200-Series Intelligent Network Recommendation*. Number Q.1200. International Telecommunication Union Standardization Sector, Geneva, October 1995.
- [4] J. Gastaud. *Conception de SIB (Service Independent Blocks) pour les Réseaux Intelligents*. Mémoire de fin d'études - ENST - Juillet 1999.
- [5] ITU-T. *Intelligent Network: Q.1210-Series Intelligent Network Recommendation Structure*. Number Q.1210. International Telecommunication Union Standardization Sector, Geneva, October 1995.
- [6] Hassan Nasreddine, Anisse Idir, and Simon Znaty. Moore: Méthode et outils orienté-objet pour la création de service reseau intelligent. In *GRES'99*, 1999.

MULTI-AGENT SUPPORT FOR MODELLING CO-OPERATIVE WORK

Mihhail Matskin

Department of Computer and Information Science, Norwegian University of Science and Technology, N-7491 Trondheim, Norway

Key words: Agents, Distributed AI, Group Communication Modelling

Abstract: We consider a development of multi-agent support [5,6] for modelling co-operative work in information systems. The development includes conceptual foundation of the method and its implementation issues. Basic elements of our approach are a flexible conceptual method for modelling co-operative work by identifying work participants and co-operative points and a concept of Agora as facilitator of co-operative work. An essential feature of our approach is its openness. The approach is demonstrated by an example of modelling co-operative work of a Program Committee of a conference.

1. INTRODUCTION

Development of computer networks dramatically changes a way of representation and usage of information. Instead of centralised archives and databases more and more decentralised information sources are available. A complete modelling of such distributed information sources is quite problematic.

The main problem is as follows. Most of former modelling approaches were designed for analysis of information in completely specified environments. These environments contain all sufficient information for operation at the environment. It does not mean that a completely specified environment should be only static. However changes in these environments are presented explicitly as well as actions which are also described

explicitly. In the other words our environment (“world”) is closed in this case.

In the case of decentralised information sources in the network an environment in many cases cannot be specified completely. It is problematic (if possible) to describe completely all available information sources in the network as well as to specify precisely and completely all possible actions and changes. The distributed environment (“world”) is not closed any more and, generally, there can be no restrictions for affecting the “world” and being affected by the “world”.

A constructive approach for work in an open environment could be similar to human’s way of behaviour in a real world which is also open. This approach is based on perceiving the world, having a rational model of behaviour and having intentions/motivations to be fulfilled by implementing corresponding goals. Of course, it is a schematic view to “human’s way of behaviour”, however, it emphasises autonomous, pro-active and distributed nature of exploring information sources and it is constructive in the case of open world. This approach becomes widely acceptable as an agent-based approach where agents are software entities/systems to whom users delegate some activity to be performed. For definition of agents we refer to [14], where agent is defined as an autonomous, pro-active, social and reactive system. We emphasis these features as essential ones for exploring information sources in distributed environment.

However not only information sources become distributed - control also becomes distributed and decentralised. In order to support decentralised control the role of co-operative mechanisms, such as co-ordination, negotiation and communication, increases very much. Exploring information in the network and perception of an open environment often requires co-operative efforts and global/local co-ordination, negotiation and communication activities of agents.

In this paper we consider an architectural support for modelling co-operative work between agents in a distributed information system. The paper is based on a generic architecture proposed in [5,6]. Compare to the previous papers ([5,6]), which are focused on justification of the approach, here we emphasise conceptual foundation of the approach and its implementation issues.

2. WORKING EXAMPLE

As a working example for our approach we use a simplified version of a Program Committee (PC) work modelling from [5].

In general we refer to intuition and experience of most of readers who participate in the work of a PC and we assume that PC includes a chairman (it may be co-chairman also) and PC members. PC is formed by the PC chairman by personal invitation of members after their agreement. Papers are submitted to PC chair(s) and then allocated to PC members for reviewing. Generally, PC members review themselves but they also can ask other experts to review a paper on their behalf. Critical co-operative points in the work of PC are:

- allocation of papers for reviewing,
- making agreement about paper evaluations between several reviewers,
- satisfying deadlines for reviewing and notifications of the acceptance or rejection of papers.

3. GENERAL APPROACH

The main question we would like to answer is as follows: what are possible/convenient ways of modelling co-operative work in an open and decentralised environment?

We choose agent-based approach as a general paradigm. The basic advantages of agents refer to their ability to autonomous, pro-active and social behaviour in an open environment (this is briefly considered in Section 1) as well as to ability to represent user-delegated goals and tasks. A detail practical justification of this choice is given in [5,6] and some experience remarks are presented in Section 5.

Briefly speaking we started with analysis of generic scenarios of co-operative work from different perspectives: functionality, data sources and co-operative work support. The functionality and data sources mostly relate to user's view of a system and the co-operative work support relates to a system organisation view. Of course functionality and data are important sources for designing system organisation, however, their influence is mostly implicit. But co-operative work perspective may provide us ideas for system organisation explicitly. As basic elements of such organisation we identify co-operative mechanisms and participants. In our case co-operative mechanisms include co-ordination, negotiation and communication and participants correspond to agents.

The main idea of our approach is as follows. We consider a co-operative work as a set of communications, co-ordinations and negotiations between participants (agents). In this case modelling of such work assumes identification of communication, co-ordination and negotiation points between participants in the co-operative work and providing support for

these co-operative points. In the case of PC work, examples of such co-operative points are as follows:

- PC formation,
- allocation of papers to PC members,
- allocation of papers to reviewers (if different from PC members),
- making agreement about evaluation of a paper (if discrepancies exist),
- satisfying deadlines for reviews and notification of the acceptance or rejection.

Examples of participants in the case of PC work are PC members and other reviewers represented by agents in a computer environment.

In order to support work in the co-operative points we introduced a notion of Agora [5,6]. Agora is an area or meeting place where agents can advertise themselves and establish a common context. In order to be able to use Agora, an agent should be registered. Registration of the agent means pointing out it's name and public information about it's properties, which may include agent's goals, tasks it is interested in, tasks it can perform etc. In addition to advertising information about agents, Agora contains information about the problem or situation it should support, common knowledge, references to related Agoras, organisational structures of co-operatively solvable problem and knowledge about information sources. Hence, a common work context can be established by explicit representation of the contents of Agora and by registration of its agents.

More generally, Agora is a facilitator of co-operative work (co-operation, negotiation and communication) in a way similar to KQML facilitators which facilitate only communication [2]. In addition to registered participants agents each Agora has default (manager) agent and registered co-ordination or/and negotiation agents (which maintain a particular co-ordination or negotiation protocols).

In accordance to the above-mentioned the elements of co-operative system modelling are as follows (see Figure 1):

- Agents: registered software agents,
- Agoras: facilitators of co-operative work,
- Default agents (managers): system agents which maintain Agora's functioning.

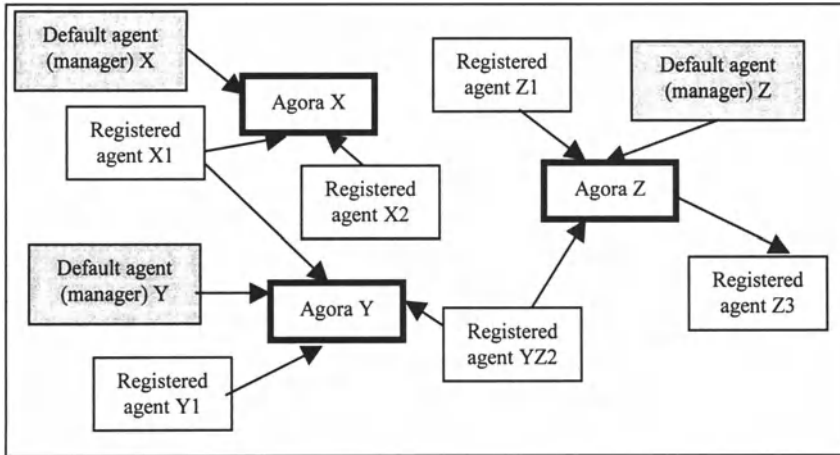


Figure 1. Agents and Agoras

4. AGORA CONTENT

Our next step in modelling co-operative work is implementation of Agora as a facilitating infrastructure for such work. In order to implement Agora decisions about Agora's content should be made. We consider the following three groups of elements in the Agora contents:

- Information about registered agents (individual agent's information),
- Information about Agora itself (group information),
- Information about other Agoras and common knowledge (surrounding information).

All Agora components have standard (default) implementation in a generic Agora shell. However the default implementations can be overridden by an Agora creator by attaching corresponding agents (in this case the components in Agora architecture become proxy components with pointers to actual components). The functional architecture of Agora is presented in Figure 2.

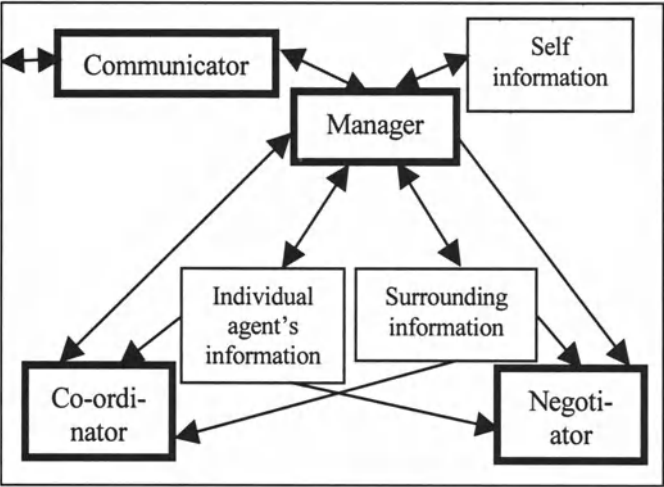


Figure 2. Agora functional structure

The communication component (Communicator) is similar to the usual agent communication block (it includes Agent Communication Language component for communication with agents and a signal-level communication component for communication with environment) and we assume a traditional solution based on KQML/FIPA [2,12] for it. We also assume that co-ordination and negotiation components (Co-ordinator and Negotiator) implement some co-ordination/negotiation protocols (e.g. [11]).

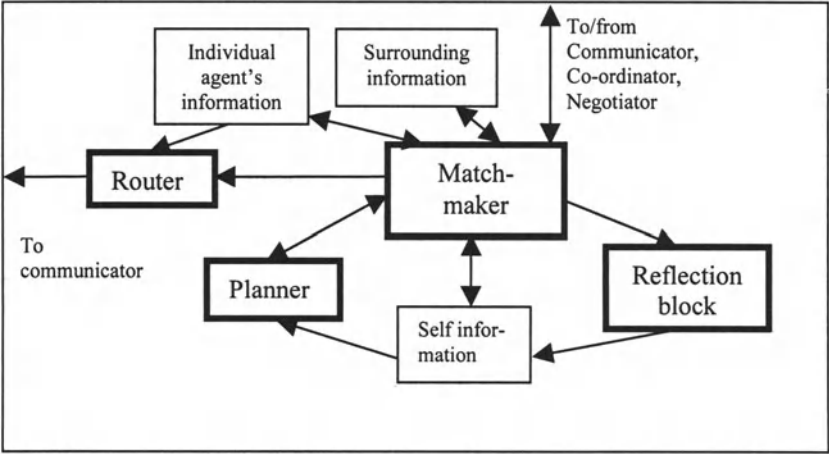


Figure 3. Agora manager

The manager is a central element in the Agora architecture and it contains the following blocks (see Figure 3):

- router - if message which is sent to the Agora should be processed by another agent then router redirects it to the corresponding agent,
- matchmaker - matches dependent/relevant information sources in Agora structure and decides what to do if matching succeeds/fails,
- planner - makes plans for action based on self information in Agora,
- reflection block - updates self information by observation of matchmaker work (more details about the reflection block can be found in [7]).

4.1 Agora information about registered agents

Agora needs to store different information about registered agents. This information includes:

- agent name and address,
- goals - they can be single (expressed by a word), complex (expressed by an expression like *finish(reviewing(before(28.02.99)))*) or multiple,
- agent's beliefs, desires and intentions - this refers to quite common for agents BDI-architecture [8],
- tasks - they are divided into two sets: tasks the agent can perform and tasks the agent is interested in,
- friend agents - other co-operating agents.

The information about agents can be used for message sending or for matchmaking. Matchmaking is an important operation on Agora content and it can be performed on tasks, goals, beliefs, desires and intentions. The purpose of matchmaking is to find another agent in Agora who can help in problem solving. Matchmaking can have different level of complexity from search for keywords to complex deductive procedures.

In the case of PC work modelling PC-Agora is created and each PC member is represented by an agent. The agents are registered in the PC-Agora where they present their goals (e.g. reviewing before deadline), tasks they can perform (e.g. papers/subjects they can review), tasks they are interested in (e.g. papers/subjects they are not experts in and where they ask some other experts to be reviewers) and friend agents (e.g. other agents who help them in reviewing). Papers to be reviewed can also be presented as agents which specify area of expertise of the papers (e.g. by keywords) and tasks they are interested in (e.g. to be reviewed by experts). However in our simplified example we don't consider papers as pro-active agents but rather as passive information sources.

4.2 Information about Agora itself

Agora information may include:

- organisational structure of agents participating in co-operative work including roles they may play in common work,
- Agora's goals and tasks – they may differ with goals and tasks of some registered agents,
- protocols for communication - in case of KQML/FIPA-like communication it may be a set of performatives which are accepted by the Agora,
- among others there should be default (manager) agent and at least one co-ordination or negotiation agent, which support corresponding protocols,
- information about current activities (own and other registered agents) including current tasks, goals etc.,
- a list of registered agents.

The above information can also be used for matchmaking and for advance interaction of agents with Agora. For example, if a role of an agent in Agora's organisational structure is restricted to reviewing only then agents having other roles will not be allowed to register themselves at the Agora. If we present papers as agents then matchmaking of areas of reviewer's expertise and paper's subjects can be done for allocation of papers to reviewers.

4.3 Information about other Agoras and common knowledge

Agora can contain information about related Agoras and about common information sources and knowledge.

In the case of PC work, related Agoras are those created by PC members for organising reviewing their papers and common information sources are, for example, papers submitted to reviewing.

Common knowledge is usually represented as ontology.

5. PUTTING PIECES TOGETHER OR LET US RUN OUR EXAMPLE

We briefly considered basic features of the approach and its support by Agoras. Now we consider running of our working example (PC work) in accordance to the approach and present experience remarks. Some elements

of performing the example are distributed around different chapters of the paper, however, here we try to put them together.

The first step in modelling PC work is making decision about participants of the co-operative work or in the other words – about agents which will inhabit our system. There are the following two possibilities: PC members can be represented by agents and/or papers for reviewing can be represented by agents. Our first decision is to restrict a set of agents to PC members agents only (we do that for simplicity only but there are no restrictions for other decisions).

The next step in our approach is identification of co-operative points in the co-operative work. Some of them (we consider only the basic ones) are PC formation, allocation of papers to PC members and to other reviewers and discussion of paper evaluation. These co-operative points should be supported by different Agoras.

When PC work starts PC chairman creates a PC-Agora (see Figure 4), registers his/her agent at it and asks potential PC members to participate in the PC work (for simplicity we assume that this is done via email, phone, fax etc.). Those who agree to be in PC should register their agents at PC-Agora (during this registration they also specify areas of their expertise in the field).

PC chair also makes decision about a procedure of allocation of papers by registering corresponding co-ordination agent. We consider two basic way of doing allocation of papers: centralised and decentralised.

In the case of centralised allocation, a co-ordination agent is a matchmaking agent. It matches keywords in papers (or titles) and areas of expertise of PC members. Conflict resolution and general constraints can be embedded into the co-ordination agent.

In the case of decentralised co-ordination agents, more advance protocols can be used. We consider only one of possible protocols: Contract Net Protocol (CNP) [11]. In this case co-ordination agent is a manager who announces work to potential contractors (PC members) as a list of papers to be reviewed. PC members submit their bids for doing this work by selecting papers they wish to review from the list and attaching priority numbers (prices) to the selected papers. Co-ordination agent analyses bids and awards papers to the higher bidder. In case of collisions, a negotiation between corresponding PC members agents happens (this may be supported by creation of a temporal negotiation Agora for these agents as in Figure 4).

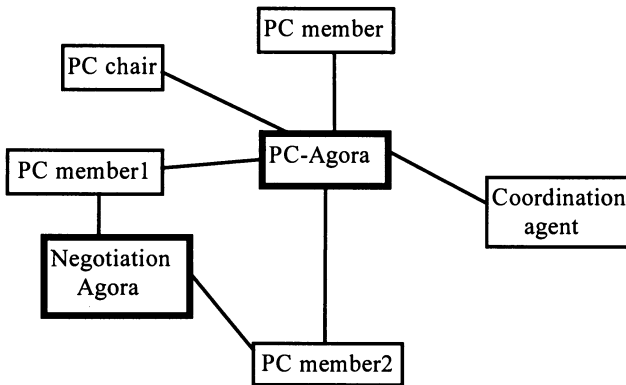


Figure 4. PC formation and allocation of papers

Next, PC members may create their own Agoras (see Figure 5) if they ask other experts to help them in reviewing - this process is similar to above-described procedure of PC formation.

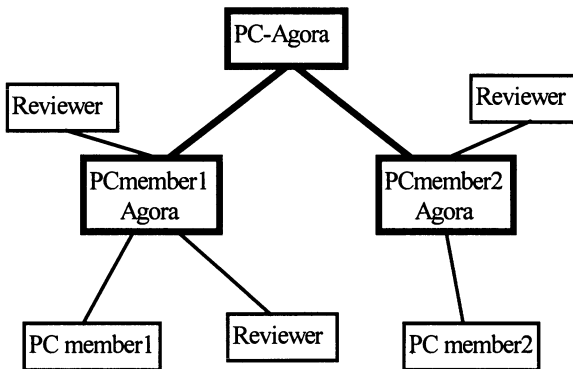


Figure 5. PC members Agoras

After that PC chair can register another co-ordination agent to the PC-Agora which provides awareness functions for notification of the deadline for reviews.

PC members submit their reviews to a co-ordination agent who analyses them and try to find discrepancies in evaluation of papers. If discrepancies are found then a negotiation between reviewers should start. In order to

support such negotiation a temporal Agora is created (see Figure 6) and a negotiation agent for support of particular negotiation protocol (for example, Monotonic Concession Protocol (MCP) [9]) is registered at it.

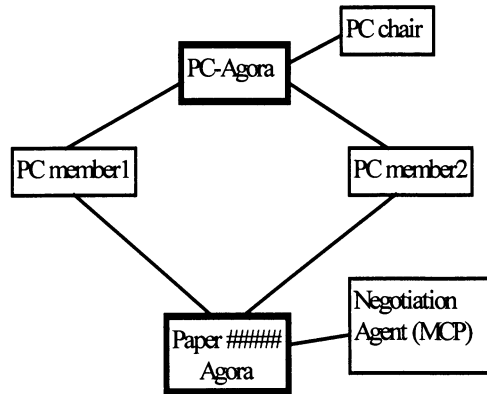


Figure 6. Discrepancy resolving

From the above example and our experience in using Agora approach we identify the following basic benefits of the multi-agent approach to co-operative work:

- Agents add speed and volume into process. This means that agents may react faster and, what is more important, in a more guaranteed way than humans they represent. For example, it is important for a fair allocation of papers to PC members to react promptly to many requests when collisions happen. However humans often do not reply to emails in time or ignore requests they think to be irrelevant – this may make the whole process of negotiation meaningless. By added volume we mean, for example, ability to more detailed negotiation via a large number of interactions with partners which are often ignored by humans.
- Automation of non-trivial time-consuming stages of the problem-solving process. In our case, conflict resolution (for example, in case of discrepancies in paper evaluation) is a critical and time-consuming stage. Multi-agent approach allows provide an automated support to such stages by means of utilisation of negotiation and co-ordination techniques.
- Personalising of software. Because of agents always represent their creator in a computational environment they should inherit user's goals, tasks and profiles and this enforces creation of personalised software.

- In the case of our working example the Agora approach may result in a higher fairness of paper allocation and in a higher quality of discrepancy resolving because of more detailed and personalised decisions.
- Agoras give a practical support for negotiation and co-ordination in the multi-agent system by providing an infrastructure and templates for co-operative points. For example, standard types of Agoras can be provided for PC work support, for group learning, for project managing etc. This means that Agoras provide not only support for a particular co-operative work but also accumulation and re-use of knowledge about such support.

6. RELATED WORKS AND CONCLUSIONS

Much research has been done, from different perspectives, in the areas of Distributed AI [4] and Computer Supported Co-operative Work (CSCW) [10] in order to support co-operative work. However most of the works consider negotiation, co-ordination and communication in co-operative work separately and do not pay enough attention to their integrated support. We try to overcome this in the case of Agora approach.

An interesting concept of communication facilitators was proposed in [2]. However our approach differs with it in the following moments:

- we allow dynamic creation and destroying of Agoras,
- we allow more flexible matchmaking by customisation of matchmaking components via possible overriding,
- functionally Agora is oriented to facilitation of different elements of co-operative work rather than only communication as in [2],
- we allow interrelated Agoras.

Another well-known approach – blackboard architecture [3] – is usually compared when Agora architecture is presented. However Agoras differ with blackboards in the following:

- we use registration of participants and do not allow global access,
- Agora allows flexible creation and it has more flexible control unit,
- Agora is oriented to co-operative work support and it always contains negotiation and/or co-ordination agents,
- Agora has richer content than blackboard and we consider our architecture to be higher level than traditional blackboards or hierarchies of blackboards [1].

Finally, we summarise the basic original features of the Agora approach as follows:

- we propose a flexible, conceptual method for modelling co-operative work by identifying work participants and co-operative points,

- we keep a proper level of abstraction in the multi-agent architecture (not too low and not too problem-oriented),
- we propose a concept of Agora as facilitator of co-operative work,
- openness is an essential feature of our approach (all basic elements of the considered co-operative work support - negotiation and co-ordination protocols, default services etc. - can be redefined by the user or agent).

A current implementation of the Agora architecture is done in Java and KQML (supported by the JATLite package [13]) is assumed to be a default communication protocol for agents.

ACKNOWLEDGEMENTS

This work is partially supported by the Norwegian Research Foundation in the framework of the Distributed Information Technology Systems (DITS) program and the ElComAg project. We would like to thank Monica Divitini and Sobah Abbas Petersen for fruitful discussions during the early stages of this work and Tom Arne Asbjørnsen and Reza Tahriri for their work on implementation of the Agora architecture.

REFERENCES

1. P. R. Cohen, A. Cheyer, M. Wang and S. C. Baeg. *An Open Agent Architecture*, in *Readings in Agent*. M.N. Huhns and M.P. Singh, Editors. 1998, Morgan Kaufmann Publishers: San Mateo, CA. p. 197-204
2. T. Finin, Y. Labrou, and J. Mayfield, *KQML as an Agent Communication Language*, in *Software Agents*. J.M. Bradshaw, Editor. 1997, AAAI Press/The MIT Press: Menlo Park, CA. p. 291-316.
3. B. Hayes-Roth, *A Blackboard Architecture for Control*. Artificial Intelligence, 1985. 26 (3): p. 251-321.
4. G.M.P. O'Hare and N. Jennings, eds. *Foundations of Distributed Artificial Intelligence*. 1996, John Wiley & Sons, Inc.
5. M. Matskin, M. Divitini and S. A. Petersen. *An Architecture for Multi-Agent Support in a Distributed Information Technology Application*. International Workshop on Intelligent Agents in Information and Process Management on the 22nd German Annual Conference on Artificial Intelligence in Bremen (KI-98), TZI-Bericht Nr. 9, Germany, September 15-17, 1998, pp.47-58.
6. M. Matskin, M. Divitini and S. A. Petersen, *Agora: a Multi-Agent Support for Distributed Information Technology Applications*. NIK'98, Tapir, November 23-25, 1998, pp. 281-292.
7. M. Matskin and E. Tyugu, *Shells for Multi-Agent Applications (An Architecture for Agents and Agoras)*. Technical report, NTNU, IDI-nr. 4/99, ISSN-NO: 0802-6394, February 1999, 43 p.

8. A. S. Rao and M.P. Georgeff, *BDI-agents: from theory to Practice*. ICMAS'95 - first International Conference on Multi-Agent Systems. 1995. San Francisco, CA, USA, 12-14 June 1995: AAAI Press.
9. J. Rosenchein and G. Zlotkin, *Rules of Encounter: Designing Conventions for Automatic Negotiation Among Computers*. MIT Press, Cambridge, Massachusetts, 1994.
10. K. Schmidt and L. Bannon, *Taking CSCW Seriously: Supporting Articulation Work*. CSCW Journal, 1992. 1(1-2): p. 7-40.
11. R. Smith and R. Davis, *Frameworks for Co-operation in Distributed Problem Solving*. IEEE Transactions on Systems, Man and Cybernetics, 1981. SMC-11(1).
12. URL: <http://fipa.com.cotec.jp/proxy/fipa/>
13. URL: <http://java.stanford.edu>
14. M. Wooldridge and N. Jennings, *Intelligent Agents: Theory and Practice*. The Knowledge Engineering Review, 1995. 10(2): p. 115-152.

THE USE OF MOBILE AGENT BASED SYSTEM AS A SOLUTION FOR EFFICIENT MANAGEMENT OF TELECOMMUNICATION NETWORKS AND SERVICES IN A MULTI-DOMAIN ENVIRONMENT

Nazim Agoulmine¹, José Neuman de Souza²,
Jorge Luis de Castro e Silva³, Arnaldo Nunes da Silva⁴

¹*Laboratoire PriSM, University of Versailles, E-mail: nazim.agoulmine@prism.uvsq.fr*

²*Department of Computer Science, Federal University of Ceará, E-mail: neuman@ufc.br*

³*Department of Technology, University of Fortaleza (UNIFOR), E-mail: jlcsilva@ufc.br*

⁴*Department of Informatics, Federal University of Pernambuco, E-mail: ans@di.ufpe.br*

Key words: Management by delegation, Shared Management Knowledge, Co-operative Objects, Distributed Objects, Mobile Agents

Abstract: The liberation of telecommunication services provision worldwide will require an intensive co-operation between telecommunication operator to provide new types of end to end telecommunication services. However, the heterogeneity between service management systems is critical and makes an end to end service management very complex. Thus, it is important to define mechanisms that permit to hide this heterogeneity and permit the exchange of management information between domain and to distribute the control over various underlying sub management systems. This work analyses the use of mobile agents and co-operative remote objects as forms to implement the co-operative management model that permit delegation and co-operation between management domains. In fact, delegation paradigm and mobile agents technology provides a powerful framework that helps the interoperability between telecommunication management systems for the support of end to end service management.

1. INTRODUCTION

The goal of this paper is to present a mobile agent based architecture to support multi-domain co-operation for the purpose of end to end management of telecommunication services. A special attention is given to the service trading mechanisms that allow value added provider to discover new basic service for the deployment of global advanced telecommunication services such as broadband virtual private networks service or Tele-Education service.

This work is mainly influenced by contemporary technological development from FIPA, OMG-MASIF, OSI (RM-ODP) [9], OMG (OMA, CORBA) [14], TINA-C [1], NM-Forum [11], Eurescom [21].

The idea of using mobile agent is motivated by the fact that centralised approach, such as OSI/CMIP [10] and IETF/SNMP [17] are not efficient enough when dealing with service level management. In the classical centralised approaches, management data are maintained by agents located in the managed devices and collected and analysed in a central way by a manager. The centralised approach for networks management has serious problems such as:

- The excessive traffic in the network between the manager and the agents and consequently the manager bottleneck. It involves an unnecessary amount of transmission between the manager and the agents in function of all the data being taken by the manager;
- The low performance of the network and the management system in function of the high traffic;
- The operator is overloaded with the amount of data and the number of alarms and events that request its attention;
- The lack of flexibility requested by the heterogeneous environment due to the increase of the services generated by the managements of the several networks (ATM and other).

The idea of this approach is to delegate management activities to co-operative management systems by mean of mobile agent. This approach was initially proposed for network management by mean of elastic agents and management by delegation paradigm [6][7]. The management by delegation can be seen as a mixture of the paradigm of the evaluation-remote [19] (where the delegated agents codes are sent to elastic servers) with the client-server paradigm (where the manager controls the delegated agents execution remotely). Therefore, the manager can delegate the management tasks to processes located in another machines or in the machines that shelter the own managed resources.

With the development of mobile agent technology, management by delegation found a framework where the distribution of control is naturally realised by mobile agent interactions.

Hence, the work undertaken by the OMG [16] through the standardisation of the architecture of object management, generally referenced by its main component: CORBA [14] provides the basic technology for protocol neutral interaction between management domains. The integration between CORBA and existing network management environments can be undertaken using proxy systems between SNMP and CORBA or CMIP and CORBA as defined by JIDM group (X/Open Joint Inter-Domain Management) [20].

Thus this article presents, as a proposal, an architecture, models and mechanisms that permits to deploy co-operation activities between telecommunication management systems for the purpose of end to end service management. The solution uses CORBA ORB (Object Request Broker) as the basic middleware to support the interaction between management system and the mobile agents as entities for the co-operation between heterogeneous telecommunication management domains.

2. TMN FRAMEWORK

A TMN system aims to respond to management needs concerning telecommunication networks and services management. A TMN provides management functions for these networks and services as well as the communication between TMNs. This management occurs by the exchange and the computation of information in order to assist the administrations in their role. An administration may comprise many TMNs and a TMN may as well cross many administrations. The TMN architecture allows the interoperability of networks managed separately too in order to provide inter-network services throw different administration.

TMNs interoperability and user's access to a given TMN are assumed at the service management layer where the service management OSFs (Open Systems Foundation). assume the responsibility of contractual aspects of services provided to clients as well as interfacing with other administrations and services interactions.

A set of reference points delimiting the managing function blocs inside and at the boundary and identifying the management information moving between blocs are identified by the TMN architecture [22]. The interface placed between two OSF blocs belonging to two different TMNs inside the same or in different administration domains is symbolised by the X interface.

In the actual deregulation of telecommunication, the deployment of the service and service management systems is intended to support the multi-provider and multi-customer service provision scenario. Thus, TMNs may co-operate to provide a global and end-to-end service management from the customer viewpoint. Hence, there is a need to deploy open service to support the Open Service Market, where the VASP (Value Added Service Provider) can discover, combine and enhance existing management services to target particular markets.

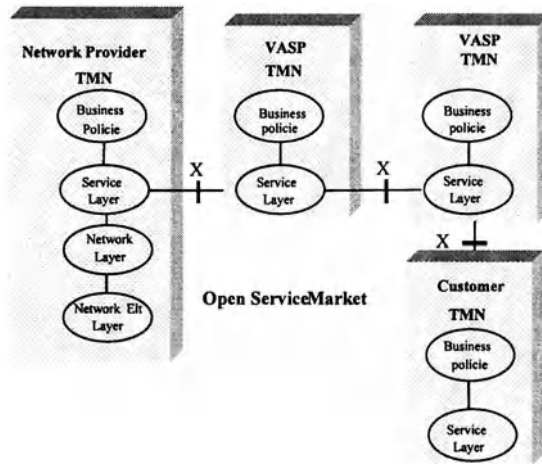


Figure 1. Interactions between Customer, VASP, PNO TMNs

Thus to support these different interaction, it is necessary to identify exactly the interfaces between the TMN at the service management layer. The information exchanged between TMNs and network users is exclusively related to TMN management functions in order to offer information concerning the usage done by these users on behalf of the network as well as a limited level of command supposing certainly that service providers supply management services to users. A common access to services and networks management application has to be provided to users in quit some conditions of security and access rules and respecting shared management schema.

3. MOBILE AGENTS

Agent technology is a growing area of research and new application development in telecommunications. Having highlighted the main requirements for security management, the intelligent agent concept seems to be a candidate approach to fulfil these requirements. What is an Agent?

Until now, there is no an internationally accepted definition of an intelligent agent concept [15][19]. The term Agent is a concept used in different area and having different meaning depending on the context. Nevertheless, different types of agents reflect a set of properties, which common among them and are described below:

- **Autonomy:** is the ability of an agent to operate without direct intervention of humans or other agents and to have some kind of control based on its internal and/or external environments;
- **Co-operation:** an Agent is co-operative and is able to have a social ability. This sociability allows an agent to interact with other agents for the purpose of performing tasks that are beyond the capability of a particular agent. This capability goes from delegation (distribution of sub-tasks) to peer-to-peer inter-working;
- **Pro-activeness:** it is the agent's ability to anticipate situations and change its course of action to avoid them. Proactive agents are capable of exhibiting goal-direct behaviours by taking some initiative;
- **Reactivity:** this kind of behaviour means that the agent reacts in real-time to changes that occur in its environments;
- **Adaptability:** is the ability of an agent to modify its behaviour over time to fulfil its problem-solving goal;
- **Intelligence:** the term "Intelligence" means that the agent is able to exhibit a certain level of intelligence priority, ranging from predefined actions (planning) up to self learning (define new actions);
- **Flexibility:** is the ability an agent should have to adapt itself to cope with the environment in which it is situated;
- **Mobility:** an Agent is mobile. It is capable of moving from one location to another in order to perform a particular task or to react to a particular event.

There are good reasons to use mobile agents (MA) in network management applications. In fact, mobile agents reduce the management traffic, because they pack a conversation among objects and they travel with the package to a target host where the interactions can occur locally. They permit to optimise the execution of management tasks as they operate in an asynchronous form and are autonomous. Thus it is possible to execute a number of tasks in the same time. Hence, they permit to hide the heterogeneous of execution environment by means of mobile agent execution environments.

Hence, they are dynamically adaptive in the sense that it is possible to customise the behaviour of mobile agents depending of the target environment.

4. DELEGATION MANAGEMENT THROUGH THE X INTERFACE

The X service interface aims to provide to management application in one TMN domain a service view of other TMN domains. The idea in this paper is not use the classical Manager/Agent interaction at this interface but rather the delegation paradigm.

In fact, in the classical manager/agent mode, the agent don't have any autonomy (except for notifications) and is statically in term of functionality. If this approach can be useful at the network element level, it is very useless at the service layer where interaction between TMN is performed.

At the service level, it is very interesting to delegate some responsibilities to other processes distributed in the same or remote TMN. Some tasks, inside of the model, can still be controlled by the central authority, but in many cases the local managers (delegated processes) can execute several tasks in an autonomous way [21].

Mobile agents provide the paradigm for the delegation of management activities.

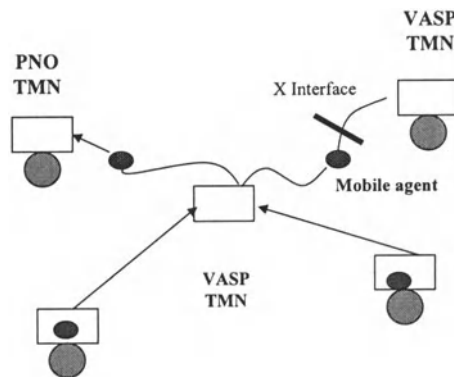


Figure 2. Co-operation by delegation

The MbD model, in contrast with the approach of centralised management, supports management model scalability. Among its advantages the following may be mentioned:

- The automation of the management functions by delegated mobile reduces the load in the operation centres;
- It eliminates or reduces significantly the need of constant polling in the network between the manager and the agents, as the mobile agent can perform directly the polling activities inside the target domain;
- New management services can be added in a dynamic way by a mean of new mobile agent to send.

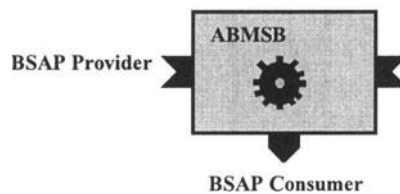
Nevertheless, management policies should be defined in order to define the authority to delegate management tasks to other TMN and to access particular management information. Each TMN should know what are its relationships with other TMN as well as the tasks it can delegate or perform.

5. ABSTRACT BASIC MANAGEMENT SERVICE

In order to support the delegation, a concept called Abstract Basic Management Service Building Block is introduced in the service management layer of the TMN. The idea behind this concept is to help to introduce the necessary abstraction at the service layer in order to hide to the service manager, developer and user the underlying complexity in term of information modelling, management protocol and service and network and system technologies. Thus, the focus at this layer will only be concerned by the service itself in terms of service modelling, service combination, service configuration, control and deployment in order to deploy quickly new enhanced services based on the interoperability of basic management service located in different TMN domains.

This concept is close to the Service Independent Building Blocks that is defined in the Intelligent Network architecture. The SIB are defined at the Global Functional Plane where the network is viewed as a single virtual machine upon which SIBs can be combined in a Global Service Logic to build an intelligent service that acts directly on the virtual machine.

The objectives is to render completely abstract to the service layer at the VASP TMN the particular specification and implementation of particular basic service in the NO (Network Operator) and Customer TMN. This approach is mandatory to define a global Open Service Market where service interaction and interoperability will be performed in an open and integrated manner.



ABMSB : Abstract Basic Management Service Block
BSAP : Basic Service Access Point

Figure 3. Abstract Basic Management Service Block

In this approach, the organisation that would like to participate to the Open Service Market should specify the management services that would like to sell to the other participants in the form of ABMSB. Thus VASP TMN will re-use existing basic service existing at the PNO (Private Network Operator) TMN and or other VASP TMN to build rapidly new enhanced management service to satisfy specific customer's demands or to develop a new type of ABMSB that could be proposed to other VASP as presented in the following figure:

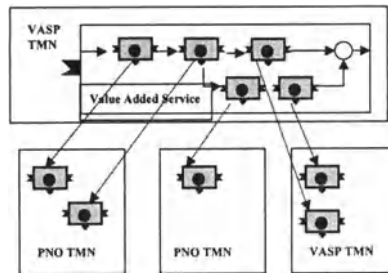


Figure 4. Basic Services Recombine and extend

This approach permits the rapid interoperability of service among a number of TMN without putting any constraints about the specification or the implementation of services inside TMN.

A particular service is built based on an interaction schema that specifies the logic of interaction between ABMSB for the purpose of an enhanced management service.

The Mobile agents provide the underlying techniques to support the effective use of ABMSB. The interactions between TMN are performed by means of mobile agent behaviour and mobile agent interaction. Mobile agent will migrate from one TMN to another to realise the physical interaction with the ABMSB. Each TMN define a set of functionality's that a mobile agent will have access in term of ABMSB in the local TMN mobile agent execution environment. The mobile agent constitutes the basic entities for component interaction and re-use. Using mobile agent, it permits to define in complete abstract manner the overall service while the realisation of the service is perfumed physically by agent transfer and agent remote activities. It is then possible to parallelise tasks, to define alternative schema and so on.

The ABMSM available at each TMN boundary can be of general purpose (e.g. configuration, alarms, performance, accounting, security) or specialised on which can be discovered dynamically.

6. ENGINEERING ARCHITECTURE

The engineering architecture defines the engineering components to deploy in order to support the previous concepts. The fundamental condition to execute an application that uses mobile code is an infrastructure. Thus in order to support interoperability, each TMN that participate to the co-operation should provide such an infrastructure. This infrastructure provides services in the transport layer that support the execution and the movement of mobile codes. Such an example of execution infrastructure is Voyager vrs.2.0.0 platform developed by Object Space company. This platform supports mobile codes based on Java. But the infrastructure can also be based on any other environment as far as it provides the basic functionality's for agent mobility, interaction, security, etc.

On the top on this infrastructure, it is necessary to develop a set of services that permit to a running agent to interact with the existing ABMSB. ABMSB can encapsulate OSI or SNMP agent to retrieve or set management information as presented in the following figure.

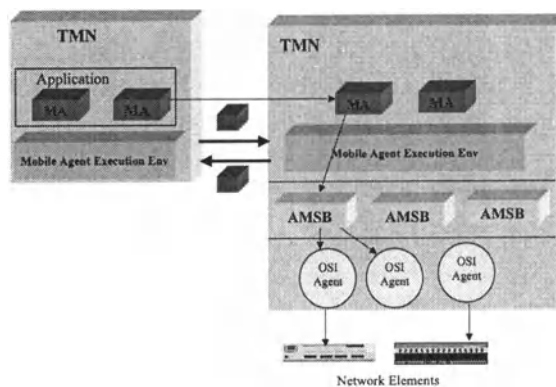


Figure 5. Mobile Agents base TMN Interoperability

But ABMSB can encapsulate more sophisticated functionalities such as VPN services, Security services, etc.

Thus, mobile agent sent to remote location has to have a minimum of knowledge and skill that will permit it to interact with the modelled remote environment in term of ABMSB. However, the interface with ABMSB is specified in term of BSAP (Basic Service Access Point), that permits to agent to discover dynamically all the services available at the particular ABMs.

7. CONCLUSION

We have presented in this paper architecture for service interoperability in a context of multi-domains TMN. The architecture permits to different TMN to co-operate by means of mobile agent interaction and Abstract Basic Management Service Blocks. ABSMSB provides the basic entity that can be reused to build an end to end service management system. And the Mobile Agent technology permits to realise the delegation of management activities in remote management systems to order to render more efficient the overall management process.

An engineering architecture was defined and a prototype is being developed using HP OpenView Java Management API and Voyager.

Hence further studies are currently undertaken in order to enhance the mobile agent interaction model and the ABMSB specifications in order to clarify the definition of service and the way they can combined in order to provide rapidly and efficiently new enhanced services.

REFERENCES

- [1] Barr, W., Boyd, T. and Inoue, Y.: "The TINA Initiative". In IEEE Communications Magazine, 31(3):70-76, 1993.
- [2] Castro e Silva, J., Silva, A. N. and Souza, J. N.; "Applying Cooperative Remote Objects and Mobile Agents Concepts for Managing by Delegation Heterogeneous Network Systems". Proceeding of the IEEE ICT'99 - International Conference Telecommunications, Cheju, Korea, June 1999.
- [3] Bieszczad, A. and Pagurek B.: "Application-Oriented Taxonomy of Mobile Code". Proceedings of the IEEE/IFIP Network Operations and Management Symposium NOMS'98, New Orleans, Louisiana, February 1998.
- [4] Borland: "Visibroker for Java 3.0 – Programmer's Guide", URL <http://www.inprose.com/techpubs/>
- [5] Castro e Silva, J., Silva, A. N. and Souza, J. N.: "Designing and Developing Distributed Applications within the CORBA and Java Environments: Practical Issues and Critical Analysis", Proceeding of the SCI'98 and ISAS'98, Orlando – Flórida, July 1998.
- [6] Goldszmidt, G. and Yemini Y.: "Distributed Management by Delegation", PhD Thesis, Columbia University, New York, NY, USA, December 1995.
- [7] Yemini, Y., Goldszmidt, G. and Yemini, S.: "Network Management by Delegation", Proceeding of the IFIP 2nd ISINM'91, Washington, D.C., USA, April 1991.
- [8] Gosling, J. and McGilton, H.: "The Java Language Environment", Sun Microsystems, Mountainview, CA 1995.
- [9] ITU-T Rec. X.901 | ISO/IEC 10746 Information Technology – Open Distributed Processing – Reference Model (RM-ODP), 1996.
- [10] ITU-T Recommendation X.701: "Information Technology – Open Systems Interconnection – System Management Overview", 1992.
- [11] NM-Forum - Network Management Forum, URL <http://www.nmf.org>

- [12] Eurescom - European Institute for Research and Studies in Telecommunications, URL <http://www.eurescom.de>
- [13] ObjectSpace: "ObjectSpace Voyager Core Package Technical Overview", URL <http://www.objectspace.com/voyager/VoyagerTechOviewNEWTAGLINE.pdf>, Dallas, USA, 1997.
- [14] OMG: "The Common Object Request Broker: Architecture and Specification Ver. 2.0, Framingham Mass., July 1995.
- [15] Pham, V. and Karmouch, A. : "Mobile Software Agents: An Overview". IEEE Communications, Vol.36, No.7, pp.26-36, Special Issue on Mobile Agents and Telecommunications, 1998.
- [16] Siegel, J.: "CORBA Fundamentals and Programming". Wiley, New York, USA, 1996.
- [17] Stallings, W.: "SNMP, SNMPv2 and CMIP: The Practical Guide to Network Management", Addison Wesley, 1993.
- [18] Sun Microsystems, Inc.: "Java Platform Documentation", URL <http://java.sun.com/docs/index.html>
- [19] Vigna, G.: "Mobile Code Technologies, Paradigms, and Applications". PhD Thesis, Politecnico di Milano, Milan, Itália, 1997.
- [20] X/Open: "Inter Domain Management: Specification Translation". The Open Group – ISNM 1-85912-150-0, February 1997.
- [21] ITU-T Rec. X.901 | ISO/IEC 10746 Information Technology – Open Distributed Processing – Reference Model (RM-ODP), 1996.
- [22] ITU-T Rec. M.3010 | Principles for a Telecommunications Management Network (TMN), Study Group IV, 1996.

MOBILE AGENTS IN IMPLEMENTATION OF A VIRTUAL HOME ENVIRONMENT

Kirsi Valtari

M.SC.EE, Leader of Information Networks group in VTT Information Technology, Finland

Key words: Mobile agents, Personal mobility, Service access

Abstract: This paper reports experiences in using mobile agents to implement a virtual home environment. The application environment is an IP-based service network, where the service access is restricted and profiled to those services, which the user has subscribed. The network is envisioned to couple various administrative and technology domains while all services may roam between the domains in order to match the users need to move around and yet connect to his personal communications environment. Agents were applied to provide the user with his personal profile in the visited domain and to negotiate with multiple service providers for the most optimal service offer. Implementation of two agents is studied and the communication trade off is reported. The benefit of using mobile agents within design is discussed.

1. INTRODUCTION

Software agents offer a new programming paradigm, which raises a lot of expectations to tackle problems common in information technology and telecommunications. These challenges include engineering of large distributed systems, optimising communication between computing nodes and supporting mobility of terminals and users.

An agent is a software program, which acts autonomously on behalf of a user (which can be a person or a computer system) or acts to achieve a given goal [10]. The roots of the agent paradigm are in the fields of artificial intelligence and human computer interaction as well as object-oriented

programming [13]. In the field of distributed artificial intelligence, the research on agent collaboration and interaction lead to multi-agent systems and also to the agent communication languages. In the human computer interaction field, the agents were seen as personal assistants of the user, who would learn the preferences and habits of their owner. If an agent has learning capabilities, it is referred to as an intelligent agent.

The concept of a mobile agent combines the characteristics of an agent and code mobility. A mobile agent is an agent that can move in the network performing various tasks [2]. The mobile agents may reduce network load and are claimed to provide asynchronous mode of communication [8]. In addition to this, they might turn out to serve as a more intuitive approach for software engineers currently working with remote procedure call paradigm. The paradigm has been proposed as a successor of client-server in designing distributed systems. The use of mobile agents requires support from the hosting nodes, which has lead to the development of several agent platforms such as Voyager, Aglet and Concordia; a nice overview of mobile agent platforms is presented in [2]. The agent platforms provide typically agent transfer mechanisms, naming and locating services and control of the agent and they could be seen as middleware which enhances Java objects into mobile objects. Voyager [9] is currently the most widely used agent platform and it has also been used in the research work described in this paper.

The agent concept is already being used as a part of the service architecture of next generation networks. In EU/ACTS programme there are roughly ten research projects carrying out prototyping and evaluation of agents within telecommunication applications such as IN, network management and service creation. The projects form a research cluster called CLIMATE [7]. Moreover, the mobile agent concept is also being studied by a number of industry-sponsored projects that investigate the use of mobile agent technology. Those include e.g., NIMA [4] (sponsored by Hitachi and GMD FOKUS) which goal is to design and implement mobile agent based solutions and functions for telecommunications network management, or MAGNA [5] (GMD FOKUS) investigating mobile agent services to be proposed for consideration as part of TINA [6]. However most of the existing projects aim to apply agents in traditional network technologies in contrast to building services framework on top of Internet. In addition concrete experiences and results on performance and applicability of agents are still needed before this new paradigm can be taken into use in operational systems.

As the Internet network keeps spreading from offices into homes and mobile phones, it has become apparent that the IP technology is the platform on top of which future network services will build on. As a next step the

telecommunications infrastructure may also be built on IP-based technology. The low cost and the endless possibilities of open technology will motivate the telecommunications players to solve the problems of security, charging and best-effort packet transfer. As user mobility has become a major requirement for networks, both fixed and mobile networks should support the end user in accessing his services and data. In a situation where all services are built on IP-technology, the user should have a service level roaming possibility over all kinds of networks both in office and in public operator administrative domains regardless of the transfer network technology.

This paper presents research work done in the context of EU/ACTS MONTAGE project [1, 3], that aims at evaluating applicability of mobile intelligent agents in telecommunications applications. The MONTAGE project has used TINA as reference architecture. In this paper the agents are applied to architecture providing service access in Internet. A specific service is envisioned to be restricted to only those who have a subscription for it. The service providers may have agreements, which allow roaming of services over their administrative domains.

2. AN ARCHITECTURE FOR MOBILE SERVICE ACCESS

In order to provide access in foreign service provider domain, a concept of Visited Service Provider (visited SP) is introduced in a similar fashion as in GSM. Mobile agents are used to support the roaming between SP domains. Additionally agents are applied in implementing dynamic service selection among competing offers from different service providers. The potential service providers who could be able to provide an instance of optimal service at a given moment are called Candidate Service Providers. The following roles are given to the service providers in this paper:

Home Service Provider(home SP) is a specific SP, to whom a user has a subscription for one or more services. The Home SP has a user specific database called user profile and a contractual position to bill the user for services used by him. It is assumed that there is a single Home SP per service.

Visited Service Provider is the initial contact point in foreign network environment. It is the service provider offered by the communication network (e.g., ATM, Internet provider or GSM). The Visited SP may also provide the actual service.

Candidate Service Provider is any service provider, who is not the visited SP and with whom the user can access services. This implies that the Candidate SP has a federation or a roaming agreement with the Home SP.

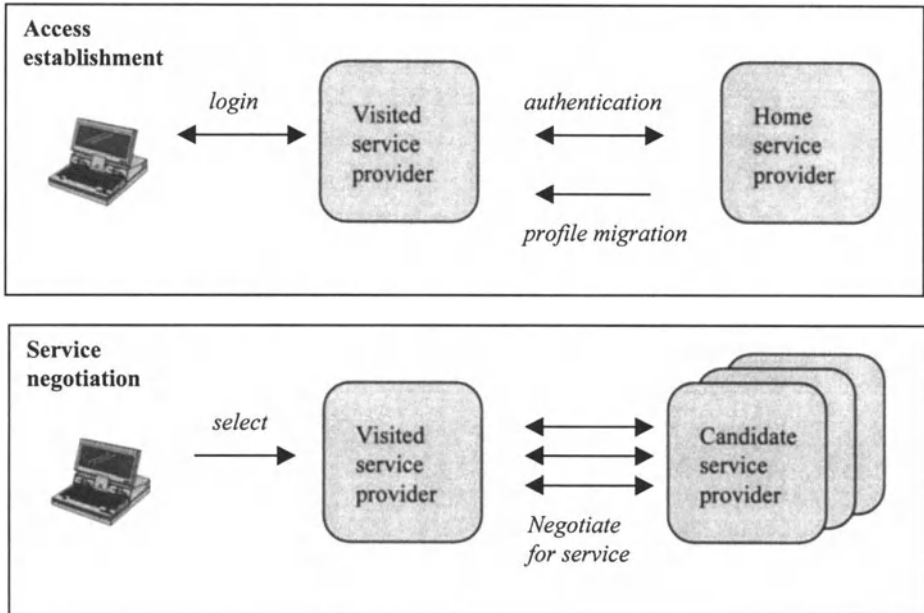


Figure 1. Communication between various participating service providers within access establishment and service selection.

Figure 1 points out the interfaces where communication is active during subsequent phases of service access and selection. The user establishes an access session with a local SP, called visited SP. The user gets authenticated through the Default SP contacting the user's Home SP. In case of positive authentication, a subset of the user's personal profile will be copied to the visited domain. A list of services the user may use in his current location is made available to him. If there are local SP's that have roaming agreements with the user's Home, the user may enjoy personalised service provided locally, while at the same time having the possibility to dynamically select a suitable service offer. Mobile agents are applied in the above scenario to provide a virtual home environment in the visited network and also to implement an efficient negotiation between the SP's to find the most optimal service offer currently available for the user.

A prototype implementation has been made in Java with Voyager [9] as the agent platform. The prototype implements the service access scenario presented in the previous section, including user agent migration and service selection. The current prototype is implemented with the following

technology: SPARC work station, Solaris 2.5.1, Java JDK 1.2, Voyager 3.0. However, in the first version of the prototype Voyager 1.2 was used together with OrbixWeb 3.0.

Figure 2 presents the hardware configuration built for demonstration and testing. It consists of three workstations (representing the SP's and a Content provider), PCs as user terminals and switched Ethernet network. The third workstation hosts a Web server acting as a multimedia content and application provider. A PC implements a user domain with Internet connectivity and a Netscape Web browser as a minimum software requirement. The user terminal does not need to support mobile agents, since they are used only in the communication between service providers. Both SP signalling and service delivery rely on IP for connectivity. The prototype is connected to Mediapoli Pilot network [12] located in Espoo, Finland.

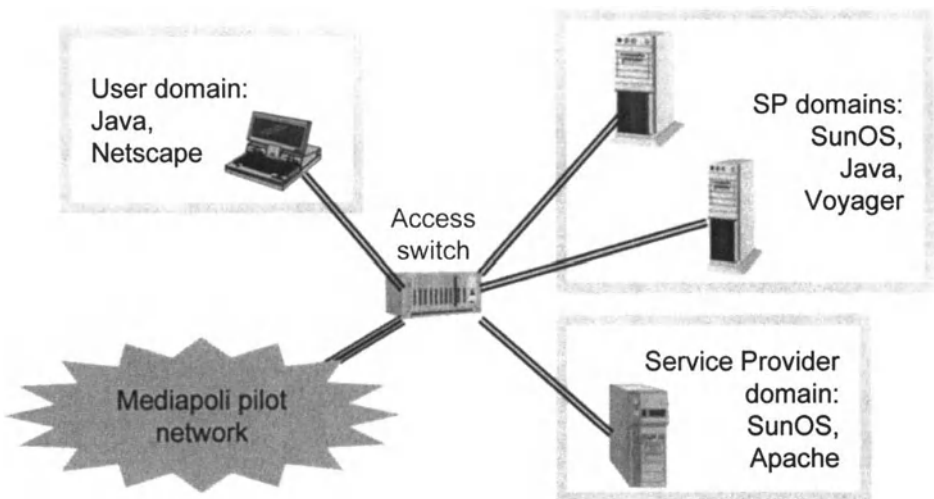


Figure 2. Network configuration used in prototyping

3. AGENT COMMUNICATION

In order to evaluate advantages of mobile agents, a comparison between mobile agent and static object implementation was conducted. A similar functionality was implemented with two different methods of communication: the traditional CORBA communication and agent communication via migration. The mobile agents studied were the User Access Agent (UAA) and the Service User Agent (SUA).

The UAA is specialised in supporting a mobile user by migrating from home domain into the visited domain where it acts on behalf of the user during the access session. The SUA migrates to potential SP's in order to

negotiate an optimal service offer. The traffic caused by the mobile agent implementation was studied and compared with static implementation of comparable functionality. The mobile agent implementation was considered in two cases. In the first case both code and data were transferred with the agent and in the second only the data moved while the code was linked from local library. The second case appears to be typical with Voyager platform, which always checks the availability of the code in target environment prior to transfer. This lead to the following measurement tests:

- Mobile agent UAA migrates to visited domain with code and data attached (mobile agent with code and data).
- Mobile agents UAA and SUA are migrating to default and visited domains carrying data only. A local implementation of the agent is used (mobile agent with data).
- The corresponding functionality is implemented using static objects and CORBA communication (static agent).

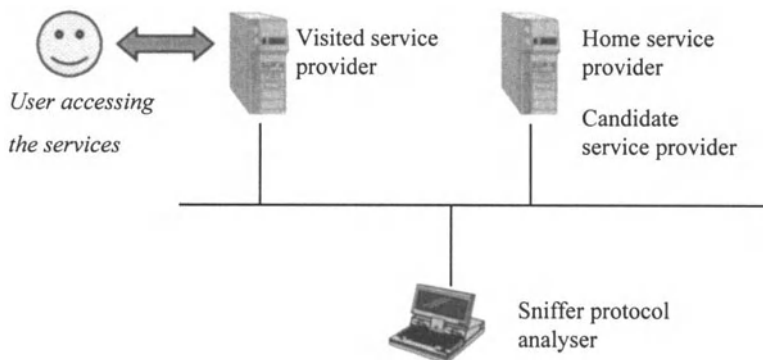


Figure 3. The test configuration used to measure traffic caused by agent migration.

In the tests the SP domains were distributed as shown in figure 3. The SP to which the user has a subscription for services (the Home SP) was located in the same workstation as the candidate to offer services (the Candidate SP). The Default SP that is supporting current access to the user was located in a second workstation. The traffic between the two workstations was recorded with Sniffer protocol analyser. The focus was on data volume of the traffic and also on the traffic pattern (number and size of TCP packets).

In the following sections the communication economics of two mobile agents and corresponding static objects are investigated. Section 3.1 focuses on the user agent UAA, which provides the remote user with his home profile. Section 3.2 gives a comparable study on service user agent SUA,

which negotiates with several SP's in order to find the most optimal service offer.

3.1 User profile migration

In the following text the problem of transferring a user profile into a foreign domain is investigated by comparing three different implementations. The logic of the scenario starts when the user logs into the network in a foreign (visited) domain and asks for an access session to be opened for him. The SP in the visited domain contacts the user's home domain asking for authentication. In case of positive authentication, the user specific profile is sent from home to the visited domain. This action was implemented in three different ways as mentioned in previous section: A) mobile agent carrying code and data, B) mobile agent carrying data, C) static agent.

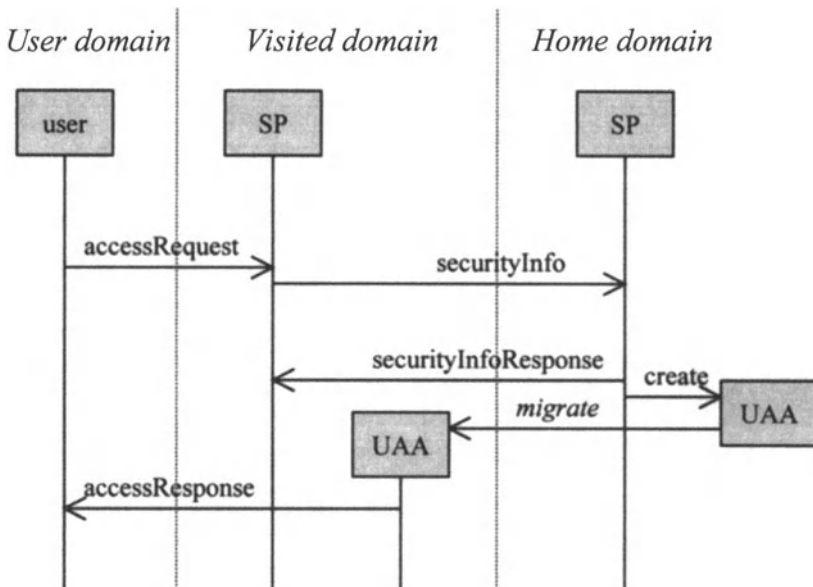


Figure 4. Service access scenario in case when the user is in a foreign (visited) network domain.

Let us first consider the two implementations where mobile agents are involved. The difference between the implementations is that in the first one both code and data migrate with the agent and in the second only data is transferred. Figure 5 presents communication study of implementations made with Voyager 1.1.6 agent platform. Column A) presents measurement

results received in case where the agent carries both data and code. In the beginning the UAA class code does not exist in visited SP domain, which forces the agent to carry code.

The A) scenario starts with securityInfo request with size of 733 bytes (which is equal in all there implementations). The Voyager platforms preparing for agent migration exchange numerous small TCP packets (24 + 19). The wasteful way of using TCP is caused by JDK 1.1 RMI implementation, which is claimed to write header data byte by byte [14]. The migration of the agent generates a TCP payload of 66 365 bytes, divided into 55 packets for transfer. In between each group of packets there is a Voyager level handshake depicted by dotted line. The overall traffic exchanged between agent platforms is 68 484 bytes.

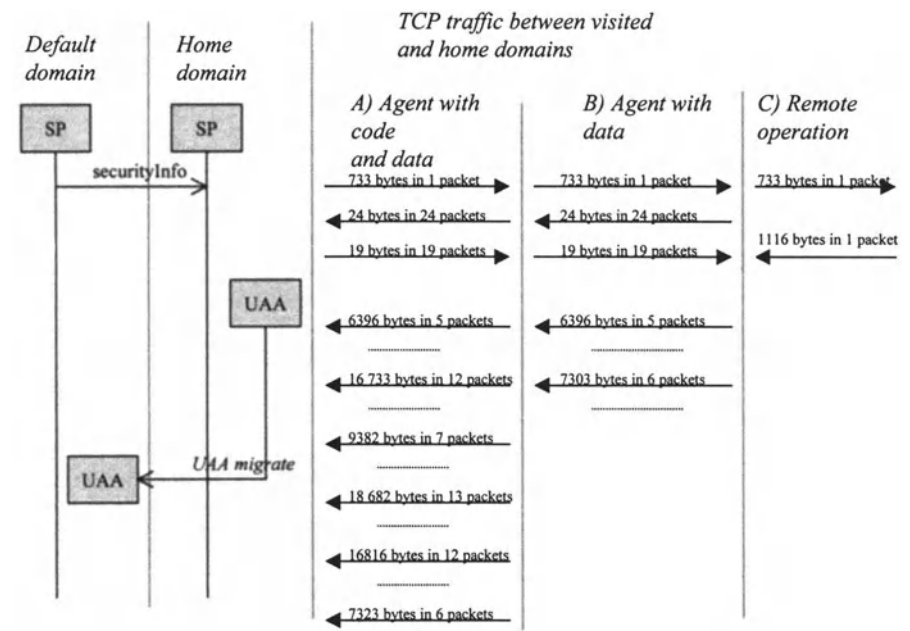


Figure 5. Traffic measurement results with three implementations.

The column B) presents a case when mobile agent UAA migrates from home domain to visited SP domain, but this time the agent carries only data. The scenario starts with securityInfo and Voyager platform specific communication as in the previous case. The migration of the agent generates a TCP payload of 13 719 bytes, splitted into 11 packets for transfer. In between each group of packets there is again a Voyager level handshake. The overall traffic exchanged between agent platforms is 14347 bytes.

When comparing implementation A and B, it is obvious that the transfer of code seems to be inefficient. The traffic volume of the agent carrying code is roughly five times bigger than the agent carrying only data.

Column C in figure 5 reflects a situation when the UAA migration has been replaced by remote operation in the implementation. However exactly the same data, the user profile, is passed as parameter. The UAA agent in visited SP is replaced by a UAA_static object. The scenario starts with securityInfo request with size of 733 bytes as in the former case. This is followed by a securityInfoResponse of size 1116 bytes, which contains the user profile data. The static implementation using remote operation appears to be far more efficient than the agent based implementation. The size of the agent carrying profile data is roughly eleven times bigger than the comparable securityInfoResponse containing user profile.

A second version of the UAA agent was implemented on Voyager 3 platform. An analysis similar to case B (agent with data) showed that the Voyager communication has substantially improved. The total communication between platforms was 8779 bytes (using two TCP connections) and the TCP round-trip number was reduced to eight.

2.1 Service negotiation

The second agent under study was the negotiating agent, SUA. The task of the agent is to find the best service offer provided by competitive service providers. In the following text the trade off of the negotiation agent is investigated as in the former section, by comparing two different implementations. The logic of the scenario starts when the user has selected a service type with a specific wish to have it delivered by the most optimal service provider. The user agent located at the visited domain will create a number of dedicated negotiating agents (SUA) and send them to potential service provider nodes to negotiate about an optimal service offer.

Figure 6 presents the message sequence in the SUA negotiation when mobile agents are used. In case of static implementation the negotiation is performed using remote CORBA calls. The sequence is designed to use agent replication which allows parallel negotiation with a vast number of candidate SP's although in the figure only two candidate SP's (and SUA agents) are visible. In the following text we will focus on the interface between visited domain and candidate SP domain in order to calculate the traffic volume exchanged during the agent based negotiation.

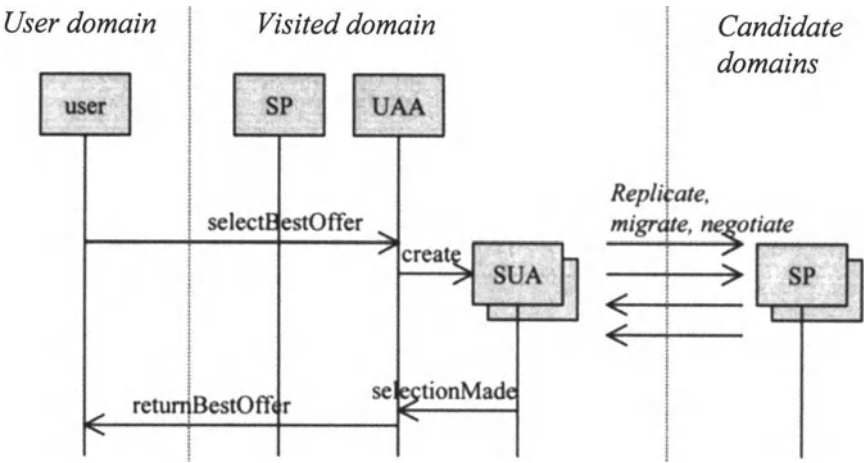


Figure 6. Negotiation for service selection.

In figure 7 column A) presents traffic analysis results when mobile agent SUA migrates from visited domain to candidate SP domain and then locally goes through a negotiation algorithm with an agent representing the retailer. The traffic relating to only one candidate SP negotiation is shown, since the negotiation with each candidate SP results in the same traffic pattern. The SUA agent in this implementation carries only data and no code. This is because the test configuration had only two workstations and home and candidate SP's were run in same computer. In such situation Voyager platform was able to detect the existence of SUA code implementation in the target domain.

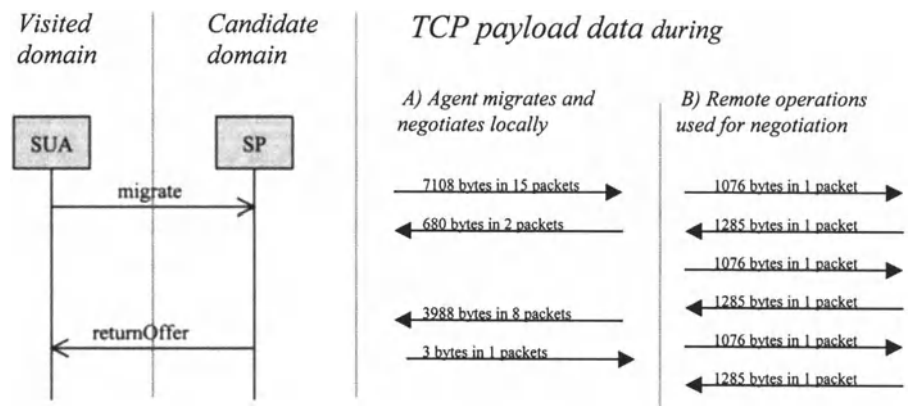


Figure 7. Traffic measurement results of SUA negotiation with one candidate SP.

The migration of the SUA agent causes a transfer of 7108 bytes from visited SP domain to the candidate SP domain. The receiving side agent platform apparently acknowledges the migration. The negotiation reply is passed as an argument in a method call and it causes transfer of 3988 bytes. The large size of the reply is caused by the user profile data, which is transferred in case the negotiation would produce a slightly differently profiled service offer. The overall traffic generated when the SUA agent negotiates with one candidate SP is 11 779 bytes. Two TCP connections are opened one for the agent migration and one for passing back the negotiation result. The overall number of TCP round-trips is 11, not including the opening or the connection. The number of round-trips could be optimised to two, if maximum packet size was used.

In figure 7, column B) depicts the static implementation corresponding to the service selection scenario in case of one SUA agent. The SUA agent is replaced by a SUA_static object, which makes a CORBA method call from visited SP domain to RA in candidate SP domain. The negotiation algorithm requires exchange of three method calls between each SUA and RA pair. In agent implementation of column A) this is implicit and not visible since it is local communication. In static implementation the negotiation leads to three remote calls in order to receive the selection result. Each of the three method calls generate traffic volume of 2361 bytes to be exchanged. It maybe easily calculated that already in case where the negotiation algorithm requires more five method calls, the total traffic volume of static implementation exceeds the volume in case of mobile agent implementation.

2.2 Discussion on measurement results

In previous sections the communication cost of two mobile agents was analysed by comparing the agent implementation with a static implementation of a similar functionality. The following figures show the total communication between the nodes during the user profile migration and service negotiation scenarios.

UAA, user profile migration (implemented with Voyager 1.1.6)

- a) mobile agent with code and data, 68 484 bytes
- b) mobile agent with data, 14 347 bytes
- c) static object implementation, 1 849 bytes

SUA, service negotiation (implemented with Voyager 3)

- d) mobile agent with data, 11 779 bytes
- e) static object implementation, 7 083 bytes

The task of the UAA agent is to roam the user's home environment into a visited domain and act on his behalf in the network side during the access. In

essence, it provides a virtual home environment for the user. A simple version of the home environment is reached when only the user profile is transferred into the distant domain. The implementation UAA c) corresponds to this case. The limitations are that the SP's must have exactly the same model of the user profile (a standardised interface) and the user must satisfy with the functionality offered by the specific SP in the current location.

A more enhanced version of home environment offers not only the profile data, but also the code and algorithms, which the user prefers. The cost of the mobile agent based implementation would probably situate between the first and second column in the table 1. This stems from the belief that part of the agent code would usually be found in the target location. Only such pieces of code, which require update or contain personalised features, would be normally migrated with the agent. This kind of behaviour is also supported by the agent platforms. Although the difference in communication overhead between mobile agent and static implementation seems big, it may become insignificant when compared with the volumes of the application data (eg. multimedia communication). Also, the overhead is not a big trade off when the availability of a preferred functionality needs to be assured. In the access scenario example, the SUA negotiating agent is not necessarily available in foreign domain unless it is brought there within the UAA agent.

The SUA agent demonstrates the benefits of mobile code in a nice way. In our test setup the SUA migrates to a candidate SP node where it negotiates locally by exchanging three request/reply pairs. This is compared with a static SUA issuing three method calls to a remotely located object. Already if the negotiation algorithm would involve five request/reply pairs, the use of mobile agents would lead to a more efficient implementation than the use of static objects from the communication economics point of view. The performance of the SUA mobile agent negotiation is optimal from the user's point of view, since all SUA replicas process simultaneously in different SP nodes and only the negotiation result is passed back to the visited SP node.

The scalability of the agent implementation studied is estimated to be reasonably good. The UAA migration consumed communication resource, but since it is not a frequent event in the network it is seen as acceptable overhead. The SUA migration scales well since by nature it converts part of the remote communication into local communication in the target node. Both agents of course require memory space and processor resources in the SP domains, which support the user mobility. A strong benefit of mobile agents is the new way to model a distributed system and especially asynchronous communication.

3. CONCLUSIONS AND FUTURE WORK

This paper has presented an implementation of service access architecture which exploits mobile agents. The agents were used to provide a virtual home environment for a nomadic user and to negotiate about an optimal service offer. The service profile of the user was carried to the visited network domain by a migrating user agent.

The communication cost of a mobile agent based implementation has been analysed and compared with a static implementation. The agent migration was reported less economical than remote operation call in all studied examples, but it was found that the migration of the negotiating agent pays off already when the agent makes five or more local method calls in the visited node. The transfer of code was found very consuming, however it is seen as a worthwhile price to pay whenever the target environment is lacking the services customised to the user. The agent platforms are still heavily evolving and this is reflected also in the implementation experiences with two platform versions.

The MONTAGE project continues to enhance the agent based service software. A user trial will be carried out in Medipoli pilot network later this year in order to evaluate the benefits of mobile agents. The security aspect of the agent based design remains for future study.

REFERENCES

- [1] Montage www page, <http://montage.ccrle.nec.de/>
- [2] Vu AnhPham, Ahmed Karmouch, *Mobile Software Agents: An Overview*, IEEE Communications Magazine, July 1998.
- [3] Henryka Jormakka, Kirsi Valtari, Andreas Kind, Dido Prevedourou, Kimmo Raatikainen, *Agent-based TINA Access Session Supporting Retailer Selection in Personal Mobility Context*, TINA-99 Conference, April 1999.
- [4] <http://www.fokus.gmd.de/research/cc/ima/projects/>
- [5] <http://www.fokus.gmd.de/ice/projects/magna.html>
- [6] TINA-C, Homepage, <http://www.tinac.com> 1998
- [7] <http://www.fokus.gmd.de/cc/ima/climate/>
- [8] Andreas Kind, Mitiades E. Anagnostou, Malamati D. Louta, Jan Nicklisch, Juan Tous, "Towards the seamless integration of mobile agents into service creation practice", IS&N Conference proceedings, April 1999
- [9] <http://www.objectspace.com/products/index.html>
- [10] S. Albayrak et al., "Intelligent agents for telecommunications applications", Proceedings of IATA 98' Workshop, IOS Press, June 1998
- [11] Michael S. Greenberg and Jennifer C. Byington, Thephany Holding, David G. Harper, "Mobile Agents and Security", IEEE Communications Magazine, July 1998
- [12] <http://www.mediapoli.fi>

- [13] Nicholas R. Jennings, Katia Sycara, Michael Wooldridge, "A Roadmap of Agent Research and Developement", *Autonolous Agents and Multi-Agent Systems*, 1, 7-38 1998
- [14] Stefano Campadello, Heikki Helin, Oskari Koskimies, Kimmo Raatikainen, "Optimizing Java 2 RMI for Slow Wireless Links", submitted for publication in *Mobile Computing and Communications Review*, Volume1, Number 2, 1999

AUTHOR INDEX

Aagesen, Finn Arve, 321
Agoulmine, Nazim, 433
Aida, Hitoshi, 177
Nunes da Silva, Arnaldo, 433
Audestad, Jan A., 135
Beier, Ines, 335
Bell, Frank, 149
Bijnagte, M.G.L., 375
Bræk, Rolv, 265, 321
Didriksen, Tor Martin, 107
Eliassen, F., 251
Emstad, Peder J., 363
Farjami, Peyman, 149
Görg, Carmelita, 149
Guan, Qi, 95
Hämäläinen, Timo, 205
Harju, Jarmo, 389
Harmen, R. van As, 219
Helvik, Bjarne E., 53, 321
Hickey, Seamus, 81
Hohl, Fritz, 45
Jaruvitayakovit, Tanun, 163
Johansen, Ulrik, 321
Luis de Castro e Silva, Jorge, 433
Jøsang, A., 375
Kantola, Raimo, 349
Karvo, Jouni, 19
Kempainen, Pasi, 279
Kleinhuis, G., 375
Koenig, Hartmut, 335
Kone, Mamadou Tadiou, 121

Leitold, Herbert, 191
Leitold, Herbert, 35
Lian, Jie, 237
Marjamäki, Harri, 349
Martikainen, Olli, 19
Matskin, Mihhail, 419
Meggers, Jens, 1
Meling, Hein, 321
Molnes, Tor Øystein, 107
Morino, Hiroaki, 177
Najm, Elie, 77, 403
Nakajima, Tatsuo, 121
Nguyen-Minh, Hoang, 219
Olsen, Frank, 403
Osland, Per-Oddvar, 363
Pagurek, Bernard, 307
Posch, Reinhard, 191
Prapinmongkolkarn, Prasit, 163
Pujolle, Guy, 27
Rafaelsen, H.O., 251
Rangsinoppamas, Naris, 163
Raza, Syed Kamran, 307
Saito, Tadao, 177
Saukkonen, Esa, 205
Silverajan, Bilhanan, 389
Simeonov, Plamen L., 293
Sørungård, Lars Sivert, 107
Neuman de Souza, José, 433
Spaniol, Otto, 1
Steensen, Sverre, 135
Tansuthepverawongse, Boonchoung, 163

Thanh, Do van, 135

Valtari, Kirsi, 445

Vignes, Sylvie , 403

White, Tony, 307

Wuwongse, Vilas, 237, 321

INDEX

- ABR service, 163
- Active networks, 45
- Active Networks, 321
- Adaptive Profile Manager, 149
- Agents, 419
- APM. *See* Adaptive Profile Manager
- Architecture, 321
- ATM confidentiality, 191
- ATM Forum Security Specification, 35
- ATM security, 191
- ATM security products, 35
- ATM security tutorial, 35
- Availability, 53
- Clos network, 177
- Configuration Management, 307
- Connection Management, 219
- Convergence, 279
- Co-operative Objects, 433
- CORBA, 307, 389
- Dependability, 53
- Design Patterns, 389
- Development life cycle, 77
- Distributed AI, 419
- Distributed Component-based Systems, 389
- Distributed meetings, 81
- Distributed Multimedia Systems, 121
- Distributed Object Computing, 135
- Distributed Objects, 433
- Distributed QoS management, 335
- ER. *See* Explicit Rate calculation
- Error persistence, 53
- Error propagation, 53
- Explicit Rate calculation, 163
- Extended Finite State Machine, 237
- Fault-tolerance, 53
- Feature interaction, 77
- Formal methods, 77
- Frameworks, 205
- Group communication, 335
- Group Communication Modelling, 419
- IN Service Creation Frameworks, 389
- INAP, 375
- Intelligent networks, 375
- Intelligent Networks, 293, 321
- Interface models, 265
- IP integration, 279
- Key agile encryption, 191
- Location Management, 219
- Logical faults, 53
- Management by delegation, 433
- MAT. *See* Mobile Agent Technology
- Max-min fairness, 163
- Mobile Agent Technology, 149
- Mobile agents, 45, 445
- Mobile Agents, 121, 135, 433
- Mobile code, 45, 135
- Mobile Computing, 135
- Mobile Services, 81
- Mobility, 135
- MSC, 237
- Multiagents, 121

- Multimedia, 251
- Multiparty interactive multimedia applications, 335
- Multiparty video conferencing, 335
- Network Convergence, 293
- Network Intelligence, 279
- Network Management, 321
- Network survivability, 53
- Nomadic Service Logic, 293
- Object-orientation, 265
- ODP, 251
- Open Distributed Computing, 135
- PA. *See* Provider Agent
- Performance evaluation, 349
- Personal mobility, 445
- Playout algorithms, 349
- Plug-and-Play, 307, 321
- Provider Agent, 149
- PSTN, 95
- QoS, 53, 121, 251
- QoS management, 121
- Reference model, 279
- Regular Expression, 237
- Reliability, 53
- Roles, 265
- SDL, 237
- Service, 77
- Service access, 445
- Service Architecture, 293
- Service models, 265
- Service platforms, 279
- Service provider access, 375
- Service Provisioning, 149, 307
- Service Roaming, 149
- Services, 95
- Shared Management Knowledge, 433
- Smart Networks, 53, 321
- SS7, 375
- TA. *See* Terminal Agent
- TCP/IP, 375
- TDM switch, 177
- Telecom and Computing convergence, 135
- Telecommunication, 321
- Telecommunication Information Networking Architecture. *See* TINA
- Telecommunications infrastructure, 279
- Telepresence, 81
- Teleservices, 321
- Terminal Agent, 149
- TINA, 219
- Typing, 251
- UIA. *See* User Interface Agent
- UMTS. *See* Universal Mobile Telecommunication System
- Unified messaging, 205
- Universal Mobile Telecommunication System, 149, 219
- User Interface Agent, 149
- Value-added services, 205
- Variable bit rate, 177
- VHE. *See* Virtual Home Environment
- Video traffic, 177
- Virtual Home Environment, 149
- Voice over IP, 95, 349
- VoIP, 95