Intelligence in Communication Systems

Edited by
Roch Glitho
Ahmed Karmouch
Samuel Pierre

Springer

ifip

INTELLIGENCE IN COMMUNICATION SYSTEMS

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 COMMUNICATION SYSTEMS

IFIP International Conference on Intelligence in Communication Systems, INTELLCOMM 2005
Montreal, Canada, October 17-19, 2005

Edited by

Roch Glitho
Concordia University
Canada

Ahmed Karmouch
University of Ottawa
Canada

Samuel Pierre
École Polytechnique de Montréal
Canada

Springer

Library of Congress Cataloging-in-Publication Data

A C.I.P. Catalogue record for this book is available from the Library of Congress.

Intelligence in Communication Systems
Edited by Roch Glitho, Ahmed Karmouch and Samuel Pierre

p. cm. (IFIP International Federation for Information Processing, a Springer Series in Computer Science)

ISSN: 1571-5736 / 1861-2288 (Internet)
ISBN-10: 0-387-29121-0
ISBN-13: 9780-387-29121-5
Printed on acid-free paper

Printed in the United States of America.

9 8 7 6 5 4 3 2 1 SPIN 11560746
springeronline.com

Preface

Communications systems are now ubiquitous. Making them more intelligent remains very challenging. A wide range of technical solutions can contribute to this intelligence. Several research areas are involved. Some examples are: architectures for adaptable networks and services; intelligent service application interface and intelligent human interaction; semantic web services and web services technologies.

Any of the research areas mentioned above include a plethora of topics. Ad hoc networks, programmable and active networks, adaptive protocols are among the topics which make the adaptable networks research area. Semantic Web is also very rich as research area. It includes knowledge representation languages, tools and methodologies; ontologies; semantic brokering; large scale knowledge management; and semantic interoperability.

Intelligent communication systems is indeed a very important domain. The goal of Intellcomm 2005 is to bring together researchers and practitioners to discuss the latest developments in the area. These developments span both the theoretical and the practical aspects of the domain. These proceedings contain the technical papers selected after a very rigorous peer review process. The papers are grouped under the following umbrellas:

- Ad hoc networks / hybrid networks / WLAN
- Security, privacy and consumer protection
- Adaptive architectures and protocols
- Flexible QoS and QoS management
- Flexible service specification, validation, searching and querying
- Service composition and Web services
- Personal, terminal and node mobility
- Programmable and active networks

We hope you enjoy the papers. We also take this opportunity to thank all the authors, members of the technical program committee, external reviewers and all the other people who have contributed to the success of the conference.

Roch Glitho

General Co-Chairs

Ahmed Karmouch, University of Ottawa, Canada
Samuel Pierre, Ecole Polytechnique de Montréal, Canada

Program Chair

Roch Glitho, Ericsson & Concordia University, Canada

Tutorial Chair

Michel Barbeau, Carleton University, Canada

Publicity Co-Chairs

Alejandro Quintero, École Polytechnique de Montréal, Canada
Haidar Safa, Am. Univ. of Beirut, Lebanon

Local Arrangements Co- Chairs

Sabine Kébreau, École Polytechnique de Montréal, Canada
Gabriel Ioan Ivascu, École Polytechnique de Montréal, Canada
Abdelhamid Ouardani, École Polytechnique de Montréal, Canada
Raymond Lévesque, Bureau des Congrès Universitaires, Canada

Program Committee

Alex Galis, UCL, UK
Kiyoshi Akama, Hokkaido University, Japan
Ali Miri, University of Ottawa, Canada
Chutiporn Anutariya, Shinawatra Univ., Thailand
Finn Arve. Arve Aagesen, NTNU, Norway
Michel Barbeau, Carleton University, Canada
Harold Boley, Nat. Research Council of Canada, Canada
Raouf Boutaba, Waterloo University, Canada
Tru Hoang Cao, Ho Chi Minh City Univ. of Techn., Vietnam
Soumaya Cherkaoui, Univ. Sherbrooke, Canada
Nigel Collier, National Institue of Informatics, Japan
Mieso Denko, University of Guelph, Canada
Phan Minh Dung, AIT, Thailand
Tapio Erke, AIT, Thailand
Dieter Fensel, University of Innsbruck, Austria
Dominique Gaiti, University of Techn. Troyes, France
Arun Iyengar, IBM Research USA
Evangelos Kranakis, Carleton University, Canada
Guy Leduc, Univ. Liege, Belgium
Thomas Magedanz, TU Berlin, Germany
Olli Martikainen, University of Oulu, Finland
Lorne G. Mason, McGill University, Canada
Riichiro Mizoguchi, Osaka University, Japan
Elie Naim, ENST Paris, France
Ekawit Nantajeewarawat, Thamasat University, Thailand
Bernhard Plattner, ETH Zürich, Switzerland
Ana Pont-Sanjuan, Polytechnic University of Valencia, Spain
Aiko Pras, University of Twente, Netherlands
Guy Pujolle, Lab. LIP6, France
Reda Reda, Siemens, Germany
R. Sadananda, AIT, Thailand
Tadao Saito, Un. of Tokyo, Japan
A. B. Sharma, AIT, Thailand
Marcin Solarski, Deutsche Telekom A.G. Laboratories, Germany
Virach Sornlertlamvanich, NECTEC, Thailand
Otto Spaniol, RWTH Aachen University, Germany
Said Tabet, Nisus Inc., USA

Do van Thanh, NTNU, Norway
Samir Tohme, University of. Versailles, France
Anne-Marie Vercoustre, INRIA, France
Naoki Wakamiya, Osaka University, Japan
Vilas Wuwongse, AIT, Thailand

Table of contents

A FRAMEWORK FOR BUILDING CUSTOMIZED ADAPTATION PROXIES *

Hana K. S. Rubinsztejn, Markus Endler and Noemi Rodriguez
Departamento de Informatica, PUC-Rio
R. Marquês de Sao Vicente 225
22453-900, Rio de Janeiro, Brazil
{hana,endler,noemi}@inf.puc-rio.br

Abstract This article presents a framework for the development of adaptive proxies for context-aware mobile applications. The framework is in charge of collecting clients' current context (device and network), and trigger the appropriate adaptations. MoCA's *ProxyFramework* offers mechanisms for cache management, as well as for adaptation management. Developers need only create their application-specific adaptations (developing *adapters* modules) and define trigger conditions, priorities and selectors. This is done by specifying rules in XML format, which define the actions to be applied at the moment of a context change. The other extension point of the *ProxyFramework* is the caching policy to be used.

Keywords: Mobile Computing, Context-awareness, Proxy, Framework

1. Introduction

A common element in the architecture of distributed applications for mobile networks is a proxy [3, 4], which intercepts the messages exchanged between the mobile clients and servers, and which is in charge of executing a number of transformations, adaptations or management functions on behalf of one or several clients, such as content adaptation, protocol translation, caching, personalization, user authentication, handover management, etc. The main advantage of using such an intermediary is to bridge the *wired-wireless gap*, and make all mobility, connectivity and context-dependent issues transparent to the application developer.

Although each distributed application for such networks has specific adaptation and transformation requirements, there are a number of common and recurrent components and interaction patterns used for implementing usual

*This project is partially funded by CNPq, Grants 55.2068/02-2 and 47.9824/04-5.

adaptation and management functions. As a means of supporting the development of proxies for several applications for mobile networks, and enhance reuse of code, we are developing an object-oriented framework that can be extended and customized to produce concrete proxy instances according to the specific application requirements.

This work is part of a wider project, where we are implementing a middleware called *Mobile Collaboration Architecture* MoCA[8], consisting of APIs and services for context-provisioning and -processing, location inference, as well as mechanisms for notifying context changes to applications. Within MoCA, the framework will be used to generate instances of customized proxies for different context- and location-aware applications. Since most of the adaptations performed by a proxy are determined by the current execution context of a mobile client, e.g. the current wireless network or Access Point being used, the quality of the wireless link, or the availability of its local resources, the *ProxyFramework* includes functions to subscribe to MoCA's context services and mechanisms to trigger adaptations according to received notifications of context changes. The current focus is on content adaptation (e.g. distillation and transcoding) and caching, which are two central issues when developing adaptive applications for mobile devices and wireless networks.

2. The MoCA Middleware

MoCA [8] is a middleware architecture for the development of context-aware collaborative applications for mobile computing. It was designed for infra-structured wireless networks, and its current prototype works with an 802.11 wireless network.

MoCA offers client and server APIs which hide from the application developer most of the details concerning the use of the services provided by the architecture (see below). The *ProxyFramework* proposed in this paper is an element of MoCA. It is a white-box framework for developing and customizing proxies according to the specific needs of the application. It facilitates the programming of distributed, self-adaptive applications for mobile networks, where adaptations should be triggered by context-change events. The proxy not only intermediates the communication between the application server and its mobile clients, but also it serves as the interface with MoCA services, as Context Information Service (CIS).

The following MoCA services are in charge of collecting and distributing context information:

- *Monitor*: this is a daemon executing on each mobile device and is in charge of collecting data concerning the device's execution state/environment, and sending this data to the CIS (*Context Information Service*) executing on the wired network.

- *Context Information Service (CIS)*: This is a distributed service where each CIS server receives and processes devices' state information sent by the corresponding *Monitors*. It also receives requests for notifications (aka subscriptions) from application Proxies, and generates and delivers events to a proxy whenever a change in a device's state is of interest to this proxy. An example of proxy's request is given by the following *Interest Expression*, {`FreeMem < 15% OR roaming=True`}. The *Interest Expression* is defined as an SQL expression using some tags, as for example, `EnergyLevel`, `CPU`, `OnLine`, etc.

- *Location Inference Service (LIS)*: infers the approximate *symbolic* location of a device, using a specific context information of this device collected by CIS: the pattern of RF signal strengths received from all nearby Access Points.

3. Overview of MoCA 's *ProxyFramework*

MoCA's *ProxyFramework* is being designed to accommodate a number of basic management and adaptation functions that an application proxy might be required to execute on behalf of each of its mobile clients. In fact, the *ProxyFramework* defines only abstract interfaces of proxy components and templates describing how these components interact. In order to implement application-specific adaptation and management functions, these components have to be extended or specialized by the application developer.

3.1 Main Components

The main envisioned components for *ProxyFramework* are described as follows:

- *Handover Management:* handles the tasks related to the migration of a client to a new network domain, such as, pre-allocation of resources at the new proxy, transfer of the client's (communication) session state, or of cached objects, to a new proxy, etc.

- *Caching Management:* is responsible for storing application-specific data, messages and user preferences of each client. This component incorporates the caching strategy, the concurrency and consistency strategy and memory management strategy (LRU, FIFO). The application developer can use a pre-defined set of management strategies, or customize some of them according to the specific needs of her application.

- *Adaptations:* implements any kind of adaptation (data compression, transcoding, summarization) of the application-specific data being transferred from the server to the client, and vice-versa, according to the client's context.

- *Message Filtering:* is responsible for filtering of messages/data to be delivered to the clients according to their context and their profile.
- *Protocol Translation:* performs the transcoding from the specific wired protocol used by the application server to any of the possible wireless protocols used for interaction between the Proxy and the client.
- *Context Management:* performs the application-specific processing of the context information, such as: subscription for notifications from MoCA's CIS, analysis of context change notifications, diffusion of context information to other proxy components, etc.
- *Service Discovery:* is responsible for finding new services, users or data, according to the user profile. The lookup function will typically access some directory services, or receive some notifications from third-party "match-making" services.

3.2 Basic Steps to Use the *ProxyFramework*

In order to instantiate a proxy from the *ProxyFramework* the application developer has to follow two main steps: first, he has to implement the adaptation actions according to the specific needs of his application; and second, he needs to create trigger rules which define when (e.g. at which context condition) these actions are to be applied.

Defining Adaptive Actions. The ProxyFramework allows to condition the execution of certain proxy actions to specific states of the application client it represents. Since these actions are specific for each application, the proxy developer must implement them.

The actions are defined by the base class `Action`, which provides some common methods, as for retrieving action parameters. Essentially, there exist two types of actions: *adapters*, which modify a message, and *listeners*, which modify some state of the proxy related to a client.

Adapter actions are executed at the moment when a message is forwarded to the client, and depending on its current context. In order to implement a specific adaptation function, the developer has to extend method `execute` of the abstract class `Adapter`. This method gets the addressee of the message to be adapted and the message *per se*, and returns the modified message, or `null`. In the second case, the original message has been discarded and consequently the flow of adaptations is interrupted.

The actions of type *listener* react to changes in the state of clients. To implement a concrete listener, it suffices to extend the base class `StateListener`, which has two abstract methods: `matches` e `unmatches`. The first is always executed when the corresponding state changes from `OFF` to `ON`, while the second is executed when it changes from `ON` to `OFF`.

Configuring Trigger Rules. The *ProxyFramework* uses a rule-based approach for determining which actions (adaptations) are needed in order to provide a better service according to the different environment conditions (context). The rule configuration should be done manually by the system administrator. With this configuration, the administrator can specify the proxy configuration for all environment conditions that the server wishes to support. The administrator can define the sequence of adaptations to apply to data and thus control the service composition, using any type of service.

The decision rules are composed by states (or contexts), that must be monitored; as well as actions which may be applied for each state. The states (or contexts) and the actions must be defined through a XML file.

```
<ProxyConf>
   <State>
      <Expression> <![CDATA[ OnLine = false AND DeltaT > 3000 ]]> </Expression>
      <Action class="proxy.listeners.DefaultCacheListener">
         <Parameter name="cacheClassName"> proxy.cache.FIFOCacher </Parameter>
      </Action>
   </State>
   <State>
      <Expression>
         <![CDATA[ CPU > 60 AND FreeMemory < 10000 ]]>
      </Expression>
      <Rule priority="1">
         <Filter>
            <!- message data type ->
            <StartWith>
               <FieldValue> <Literal>datatype</Literal> </FieldValue>
               <Literal>image/</Literal>
            </StartWith>
         </Filter>
         <Action class="proxy.adapters.ScaleImageAdapter">
            <Parameter name="factor">0.5</Parameter>
         </Action>
      </Rule>
   </State>
</ProxyConf>
```

Figure 1: Trigger Rules Configuration - XML file

Figure 1 shows an example of a *ProxyFramework* configuration file. In this example, element `State` represents a monitored state and has a single element `Expression`, which corresponds to the context *Interest Expression* that will be registered at CIS for periodic monitoring and delivery of corresponding notifications, whenever the expression switches from true to false, and vice-versa. When a change happens in either direction, the corresponding customized *listener* action will be executed. Its configuration is done through element `Action`, where it is possible to indicate the class which implements the desired action, as for example, caching with FIFO policy. Each state may have several elements of type `Rule`, which aggregate several adapters which will be executed if the state for which they were registered is ON, and a certain condition related to the message (type) or the addressee is satisfied. The condition is determined through element `Filter`, which can be configured through the use of a number of logic and other operators, such as (AND, OR, NOT, EQUAL, STARTWITH) and available selectors such as (datatype, protocol, client, communicationmode, subject). Once the filter has accepted a message, the series of adapters registered for this rule will be executed. Adapters must also be registered with a rule, using the element `Action`.

It is possible to provide parameters both to the listeners and to the adapters, and this is done using element `Parameter` (each of which has a name and a value), as shown in the example.

4. Current Prototype of the *ProxyFramework*

The *ProxyFramework* consists of a set of basic functions and mechanisms for customizing, activating and combining adaptations, for the development of application proxies. Moreover, it provides the application developer with a simple means of accessing the client's context and defining context-dependent adaptations. The *ProxyFramework* was implemented in Java and offers these facilities through the structural reuse of components that are common to all application proxies, for example those for processing context notifications.

The framework is composed of a set of concrete components (*frozen-spots*), which implement utility functions for the proxies; and interfaces of abstract components (*hot-spots*), which can be implemented according to the specific need of each application. The frozen-spots include Communication, Caching and Adaptation Management and Selectors, while hot-spots are the Cache-Policies, Adapters and Listeners, and Context Configuration.

Essentially, the *ProxyFramework* is composed of two parts: the communication sub-system and the proxy *core*. While the first implements the protocols for synchronous and asynchronous communication with clients and servers, the second is responsible for collecting the context notifications regarding the clients and managing the execution of the adaptations according to the rules specified by the application developer (c.f. Section 3.2). Due to limitations of space, in the following we will further detail only the core.

4.1 Proxy Core

In order to achieve loose coupling among the different components of a proxy, and allow for their concurrent execution, the core architecture has been structured as a set of independent elements called *Managers*, and a singular manager called *Dispatcher*, which intermediates the interaction between any pair of Managers, such as those described in Section 3.1. This way, a manager does not need a reference to all other managers it interacts with. This decoupling also facilitates the inclusion of new managers. Each manager has a private queue of messages, which are processed in FIFO order. The components of the proxy core are the following:

AdapterManager. It manages the message adapters, inspecting and modifying messages according to the specific states of the corresponding destination client. Once the states to be monitored have been defined, the proxy starts to trace the status of each state, for each client. This way, it is possible to establish a set of adaptation strategies to be applied to each message, for each client. The implementation of the specific adapters (c.f. section 3.2), the order of their execution, and the criteria for their application on each message type, are all customization points of the framework, which have to be defined/implemented by the application developer.

ContextManager. This component subscribes to MoCA's CIS according to the expression defined in the XML file (c.f. sec. 3.2) and receives messages from this context service about the current state of every client registered with the proxy. The ContextManager receives notifications from CIS (i.e. a CISMessage), whenever the interest expression (which defines a client state) flips between true and false. Essentially, a CISMessage contains three pieces of information: the client whose context changed; an identifier of the changed state; and the type of transition (i.e. ON, for a transition from off to on, and OFF, for a switch from on to off). Using this information, the state of the corresponding client is updated in the proxy. In this case, i.e. at the moment of this transition, it is possible to execute some specific actions of type listener, which modify the behavior of the proxy for the following message addressed to this client.

CacheManager. It is responsible for checking if according to the current state of a client, the messages addressed to it should be cached. This may be necessary when either the client gets (temporarily) disconnected, or the bandwidth of its wireless link falls below a given threshold. When a message for a client arrives, it verifies the state of the addressee, and then either records it in the cache, or forwards it to the AdapterManager.

The framework provides a special listener action for caching. This action is implemented through class `DefaultCacheListener`, which just activates or de-activates a given cache policy, which is passed as a parameter to this class and hence can be customized by the application developer.

The framework makes available a simple default caching policy, FIFO-Casher, which stores messages in FIFO order.

Sender. The Sender is responsible for delivering the intercepted messages to the corresponding addressee. This component implements a mechanism which ensures the ordered delivery of messages to each client.

Figure 2 depicts the logic relationship between the managers, and the message flow within the proxy core, from the moment it is received from the server until it is forwarded to the corresponding client.

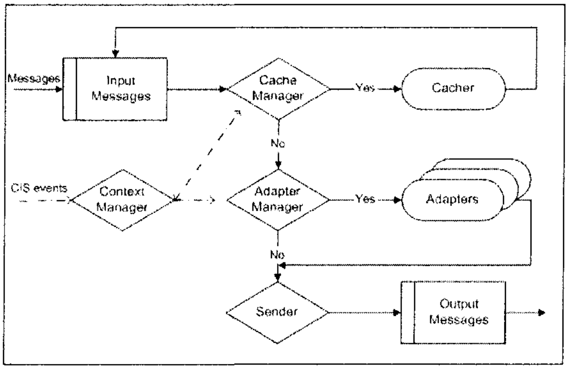


Figure 2: Message Logic Flow

Every incoming message is first inserted in the Input Message queue, and is then retrieved by the CacheManager, which checks if the message should be cached, or if it can be directly sent to the client. At the next stage, the message is sent to the AdapterManager which verifies which adaptations are to be applied to the message. After all adaptations, if any, have been applied the messages are enqueued in Output Messages, and are sent to the corresponding client in FCFS.

When caching is required, the messages are cached according to the caching policy defined by the developer. When the client's context changes, all of its cached messages return to the input queue, as if they were arriving at this moment. This is necessary due to the possibility that while some of these messages are being processed, the client's state changes, and some messages need to be cached again.

Our decision to implement the check for caching before the check for adaptation in the proxy's message flow was based on the understanding that the processing-intensive adaptations should be done according to the current client state, and only immediately before the message is sent to the client. Otherwise, the adaptations would not be effective, and hence useless.

5. A First Instantiation for Image Adaptation

The first instantiation of *ProxyFramework* was for an application that transfers and adapts images sent from a server to clients. The development of this context-aware proxy was simple and required only the implementation of some image adapters and the definition of trigger rules in the configuration XML file (c.f Section 3.2). The implemented adapters were for transforming color images into grayscale, for converting any image into JPEG with a compression quality, for scaling and for cropping.

Using our first proxy instance, we made some initial tests (using AspectJ) to evaluate the overhead introduced by the proxy. This overhead takes into account only the message management and queueing, the matching of the client state and the selection of the adaptation to be performed. It does not include the time spent on the adaptation *per se*.

In our experiments the scenario was composed by one server (source of images), one proxy and a set of clients, in which we varied the number of clients from 10 to 100. The proxy was configured with five states of interest and received images for adaptation at a rate of 2 messages per second. Each message for clients was of size 100 KB. For each set of parameters, we made 20 executions and calculated the mean value of proxy overhead. For these tests we did not cache the messages, but all the messages passed through the CacheManager, which did not act upon the messages. We executed the proxy on a 2.4 GHz Pentium 4 with 512 MB RAM.

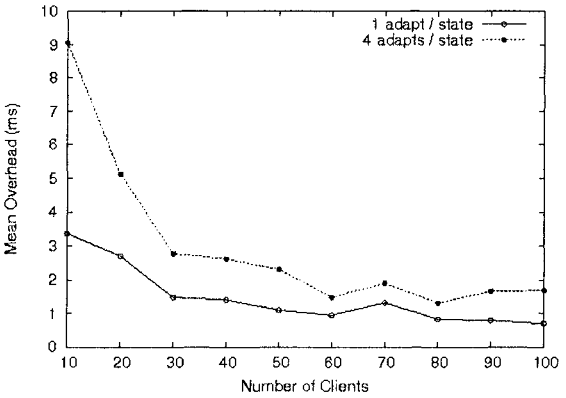


Figure 3: Number of clients x Overhead (msec)

Figure 3 shows the results of our measurements. As expected, the number of applied adaptations affects the mean latency within the proxy, since the messages stay more time in the queues waiting to be adapted.

In all curves the values for small number of clients are quite high, but this is caused by the fact that the initial Java class loading overhead is proportionally greater for fewer messages (due to fewer clients) than it is for a greater number of messages.

6. Related Work

Several other efforts have been made to develop generic proxy architectures, or *proxy frameworks*, that can be customized or extended to solve a particular problem, for example, Mobiware [1], RAPIDware [7], Web Intermediaries (WBI) [3, 6], MARCH [2] and TACC [4].

The decision of which adapters to use and when to use them can be defined in two ways: via programmable interfaces, as in Mobiware and TACC; or via rule-based configuration, as MoCA's *ProxyFramework* , MARCH and WBI. Rule-based systems are easily configured and less error prone (defining a model) than the ones based on programmable interfaces; besides there is no need to deal with intrinsic details of the framework. Furthermore, only the content provider can decide which adaptation is acceptable under different contexts, and thus, by using rules, may define the sequence of adaptations to apply to data, better controlling their composition, which is a very complex task to automate.

Comparing the two most common approaches for loading adapters, the dynamic loading of adapters, as in MARCH and RAPIDware, supports on-demand loading of adapters from an adapter repository, and provides more flexibility to the system. However, statically configurable proxies support verification of a consistent combination/configuration of adapters. In these proxies, the adapters are defined at proxy deployment time, like in WBI and *ProxyFramework.* In addition, dynamic (down)loading of adapters can be time consuming. Therefore, it is more suited for systems where context changes are not very frequent.

Comparing the systems, all of them support content adaptation, while some of them also implement caching management. Handover management is provided only by Mobiware. Concerning communication capabilities, only MoCA's *ProxyFramework* supports asynchronous (publish/subscribe) communication, which is very useful for mobile computing [5]. Context awareness is also supported by most of the frameworks (i.e. except WBI), but only MARCH and MoCA's *ProxyFramework* consider also the state of the client's devices. Our framework is the only one that supports connectivity-aware caching, where caching is automatically activated as soon as client's connectivity state changes.

7. Conclusion

As the number of applications for mobile networks increases, and their services become more complex and personalized, proxies will be used for an increasing number of specialized functions. Although each (type of) application will have specific demands for proxy based functions, we have identified a common and recurrent set of functions in proxy implementations which shall be used as the basis for developing proxies for specific needs. Based on our experience in developing some context-aware application prototypes, we felt that there is an increasing demand for flexible and extensible tools and frameworks for the rapid development and customization of proxy-based architectures.

In this paper we have presented a framework for the development of proxies for mobile computing. Our first prototype includes caching, message filtering and context-aware adaptations, since these form the core functionalities of a proxy. Our future work includes the design and development of components responsible for handover, authentication and translation for different mobile protocols. Another feature is the interaction with Location Services (as MoCA's LIS) in order to be able to implement location-based adaptations.

References

[1] O. Angin, A.T. Campbell, M.E. Kounavis, and R.R.-F Liao. The Mobiware Toolkit: Programmable Support for Adaptive Mobile Netwoking. *IEEE Personal Communications Magazine, Special Issue on Adapting to Network and Client Variability*, August 1998.

[2] S. Ardon, P. Gunningberg, B. LandFeldt, M. Portmann Y. Ismailov, and A. Seneviratne. March: a distributed content adaptation architecture. *International Journal of Communication Systems, Special Issue: Wireless Access to the Global Internet: Mobile Radio Networks and Satellite Systems.*, 16(1), 2003.

[3] R. Barrett and P. P. Maglio. Intermediaries: An approach to manipulating information streams. IBM Systems Journal 38, IBM, 1999.

[4] E. Brewer and et al. A network architecture for heterogeneous mobile computing. *IEEE Personal Communications Magazine*, October 1998.

[5] Gianpaolo Cugola and H.-Arno Jacobsen. Using publish/subscribe middleware for mobile systems. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(4):25–33, 2002.

[6] Steven C. Ihde, Paul P. Maglio, Jörg Meyer, and Rob Barrett. Intermediary-based transcoding framework. In *Ninth International World Wide Web Conference*, Amsterdam, The Netherlands, 2000.

[7] Philip K. McKinley, Udiyan I. Padmanabhan, Nandagopal Ancha, and Seyed Masoud Sadjadi. Composable proxy services to support collaboration on the mobile internet. *IEEE TRANSACTIONS ON COMPUTERS*, 52(6):713–726, June 2003.

[8] V. Sacramento, M. Endler, H.K. Rubinsztejn, L.S. Lima, K. Gonçalves, and F.N.do Nascimento. MoCA: A Middleware for Developing Collaborative Applications for Mobile Users. *IEEE Distributed Systems Online*, 5(10), October 2004.

ADAPTABLE SERVICES AND APPLICATIONS FOR NETWORKS

Josep Polo and Jaime Delgado
Universitat Pompeu Fabra. Technology Department, Psg. de Circumval·lació, 8, 08003 Barcelona, Spain {josep.polo, jaime.delgado}@upf.edu, http://dmag.upf.edu/

Abstract: New services and applications that use extensively telecommunication networks are currently developed. They need an open access to the telecommunication networks for adapting to them. An API (Application Programming Interface) that permits this objective is the Parlay/OSA API. Operators offer tools that facilitate the development of applications and services, but these tools are different, so it is not possible to develop a unique application for different operators. To solve this great inconvenience, we propose a solution to permit interoperability among different development tools.

Key words: Parlay; OSA; network; application; service.

1. INTRODUCTION

The use of network services is growing. In order to facilitate their development an API that offers many advantages is Parlay and Open Service Access (OSA). This API facilitates the services development, but it is still very complex, difficult and hard to use. Many operators and service providers offer different development tools, SDKs (Software Development Kit), that facilitate the development and implementation phases.

This document begins with an introduction about the Parlay/OSA (P/OSA) API and it shows its complexity. Then, it demonstrates how a service can be developed using those tools more easily that directly with the API. We show one application for each SDK. Later, we propose a solution to interoperability among different SDKs. We suggest a method to develop a unique service to be used in several SDKs, and finally we obtain some conclusions.

2. PARLAY/OSA OVERVIEW

Parlay and OSA (P/OSA) are an open API for communications networks. This API can support present and future networks. It provides a layer of abstraction for service developers. It enables telecom operators and service providers to offer the same services for all existing underlying networks: mobile, fixed and IP networks, without adapting the application to network specific protocols.

This API permits to obtain network-related context information, it can facilitate value-added services development, it can make network communications simpler and powerful, independently of which type of network.

2.1 Description

The P/OSA model is split into three main entities [12]:

- The client application: the application developed by a third party can access to the network features through the P/OSA interface.
- The framework: it offers support functions. For instance, security, integrity and management features.
- The services: they offer access to network features. For instance mobility, messaging, terminal management, user interaction and call control.

P/OSA has several interfaces to allow access to different network functionalities or services [12]: Framework, Call control, Data session control, User interaction, Mobility, Generic messaging, Terminal capabilities, Connectivity management, Account management, Charging, Policy management, Presence and availability management.

2.2 Application architecture

When developing a service to use P/OSA, it has three main steps:

- Authentication: before the application can use the network services, the application and the framework authenticate each other. Authentication prevents unauthorized access to the services and permits to determine the privileges and permissions to the application.
- Service selection: after the authentication phase, the application can select the service interface to use. It is usual to sign an agreement before use of the interface. The application can select the service to use.
- Service use: if the previous phases have concluded properly, the application can use the selected service.

To obtain a general idea on interface use, we shall explain only the first step. This step is common for all developing services or applications and before it can use any network service there must be an authentication step.

This step has several phases:

- The application performs an initial contact to the framework. In this first contact the application requests authentication. The framework answers with the authentication to be used.
- Then the application requests authentication from the framework.
- After the previous authentication, the framework requests authentication from the application.
- When the application and framework have authenticated each other, the application can select the service interface to use. This corresponds to the service selection step.

These phases correspond to many operations. The authentication step is a quite simple operation. The other steps are usually more complicated and large, depending on the service.

The previous authentication step description is enough to show a good idea of the use of P/OSA. From the previous paragraphs it can be concluded that the direct use of P/OSA can be very complicated, tedious, difficult and long effort. To solve this situation, few libraries exist. They shield the details of P/OSA use. They provide abstraction from the original API. Usually, these libraries are developed to use Java APIs.

3. DEVELOPMENT TOOLS

Many operators have developed tools that facilitate the effort to create applications. Usually, these tools are composed by a SDK and a simulator.

Those SDKs and simulators share characteristics. The most important are:

- Java software libraries. P/OSA defines a language-independent API and they use the Object Management Group (OMG) [6] Unified Modelling Language (UML) and Interface Definition Language (IDL) [5] that is based on the OMG's CORBA IDL. Using java libraries permits the abstraction from the CORBA.
- Partial emulation of P/OSA APIs. The systems don't support all P/OSA interfaces. They are enough for developing and testing most applications.
- They can run on an off-line way, no network connection is needed if a simulator is used. This can permit an easy application development.
- They have a similar structure. They offer a Java library, which simplify the development, but the application can use CORBA, if necessary.

To simplicity and avoid confusion, we show only two of them to obtain a general idea of their use. They are: Lucent - MiLife ISG SDK [5] (MI SDK) and Ericsson NRG SDK [1] (EN SDK).

4. SERVICE AND APPLICATION IMPLEMENTATION

Services and applications have three big parts, when using these systems:

- Initialization. In this step the application initializes all needed processes and obtains all resources for the correct operation, and interacts with the framework, which enables an application to obtain and release service managers.
- Main functionality. This is the most important part of the application. In this step the application interacts with the network to do the features for those it is developed. It uses service managers to send requests to the network, and receives responses from the network.
- Finalization. The application stops interacting with the network. The service managers are released and framework access is terminated.

All applications have the first and third phases (initialization and finalization) very similar, but the actually important step is the second one (main functionality). Each application has this phase different.

To show how to implement a service or application using these tools, we explain the main steps necessaries for a simple sample application. We think that it is enough to have a good idea how a service can be developed.

4.1 Sample application

The sample application demonstrates the framework. It obtains and shows available services defined in the network.

The basic steps of each three main parts of this application are:

- Initialization.
  - *Prepare the resources and add components and read the configuration*
  - Initiate the access to the framework
- Main functionality.
  - Obtain the names of all available service types
  - Obtain a service manager
- Finalization.
  - Releases a service manager by terminating its service level agreement
  - Release the system resources obtained in the initialization phase

Following sections show the most important operations to use each SDK.

4.1.1 Lucent - MiLife ISG SDK

We focus on how to use the MI SDK to obtain the desired P/OSA functionality. We only describe the operations that perform those functions.

The interfaces use (see section 2.2) has three main steps: authentication, service selection and service use. The first step, authentication, correspond to the initial operations, before the service use.

The first step can be done using the class `FrameworkAdapterFactory`. A framework adapter can be created using this class. The sentence is:

```
FrameworkAdapterObject =
   FrameworkAdapterInstance.createFrameworkAdapter
                    (validAuthenticationCredentials);
```

The `FrameworkAdapter` is an interface that defines the framework classes. A reference to this interface can be obtained by the previous method. The `validAuthenticationCredentials` are parameters that have been set to proper values.

This single operation performs that first step, authentication. The needed operations, if P/OSA is used directly, are related in section 2.2. It can be seen that this class performs most work and simplifies the application development.

The following operation is to obtain the available services. This can be done using the method `listServices`. The sentence to use is:

```
String[] object =
             FrameworkAdapterObject.listServices();
```

This method returns a list of available services.

The next operation is to obtain the service adapter. This operation can be done using a method specific to the wanted service. For instance, if we want to select the User Location service, the sentence to use is:

```
serviceAdapter =
FrameworkAdapterObject.selectUserLocationService();
```

This service permits to obtain the geographical location of users, but if we want to select the Messaging service, to manage, send or receive messages, the sentence to use is:

```
serviceAdapter =
   FrameworkAdapterObject.selectMessagingService();
```

As we can see, each service is selected using a specific method.

Then the service can be used.

When the services are no longer needed, it is necessary to release resources. This operation begins with the method `destroy`. The sentence to use is: `serviceAdapter.destroy();`

It allows this adapter to clean up when it is no longer used.

The releasing operation continues using the method `endAccess` that releases resources used by the framework. The sentence to use is: `FrameworkAdapterObject.endAccess()` . It ends the access session.

Finally, the method to use is `destroy`. This method allows the `FrameworkAdapter` to clean up when it is no longer used. The sentence to use is: `FrameworkAdapterObject.destroy();`

4.1.2 Ericsson NRG SDK

In this section we focus on how to use the EN SDK to obtain the desired P/OSA functionality.

The first step can be done using the class `FWproxy`. This class is for handling interaction with the framework, which enables an application to obtain and release service managers. The sentence to use is:

```
FwproxyObject = new Fwproxy(configuration);
```

The `configuration` has been set previously to proper values.

The following operation is to obtain the available services. This can be done using the method `listServiceTypes` of the class `FWproxy`. The sentence to use is:

```
String[] object =
                        FwproxyObject.listServiceTypes();
```

The `FwproxyObject` is the previously object created. This operation returns the names of all available service types.

The next operation is to obtain the service manager. This operation can be done using the method `obtainSCF` of the class `FWproxy`. The sentence to use is:

```
IpServiceObject =
                FwproxyObject.obtainSCF(UserLocation);
```

The `UserLocation` is the service to use. This operation returns a service manager. In this example, this service allows application to obtain the geographical location of users. Or if we want to manage, send or receive messages, for instance, then the sentence is:

```
IpServiceObject = FwproxyObject.obtainSCF(Message);
```

When the service is no longer needed, it is necessary to release resources. This operation begins using the method `releaseSCF` of `Fwproxy`. The sentence to use is:

```
FwproxyObject.releaseSCF(IpServiceObject);
```

The releasing operation continues using the method `endAccess` that releases resources used by the framework. It belongs to the class `FWproxy`. The sentence to use is: `FwproxyObject.endAccess();`

The method to releases all resources is `dispose` of the class `FWproxy`.

```
FwproxyObject.dispose();
```

5. INTEROPERABILITY

In the previous sections we have seen that those SDKs are different. Due to this reason, a unique application cannot be used with different SDKs. It is necessary to develop different applications, one for each SDK. Those applications are nearly equal except the use of P/OSA interface through SDK java libraries. To isolate the developing application from the particular SDK used we propose to use an interface that hides the particular implementation of each SDK. This interface is in a preliminary status. In this section we present the first results, that where introduced in [8].

This interface shows to developing application unique classes to access P/OSA interfaces, independently of which library used. In following sections we show a possible implementation of this interface.

This interface can consists in a set of unique classes with different implementations. This can be done in java using abstract classes. The new developing application uses the abstract classes and depending on which SDK we want to use, we utilize an implementation or another. All particular details are encapsulated on each implementation.

Following sections show this interface applied to the framework. There is an abstract framework class and two no abstract framework classes, one for each SDK.

5.1 Framework abstract class

The abstract classes contain a definition for all SDKs. These classes have the methods to access the P/OSA interfaces to be used by the application. Figure 1 shows a possible abstract class for the framework that contains the definition of few methods, it is not complete. They are some of the initialization and finalization steps methods. None of then is implemented, because the implementation depends on the particular SDK used.

```
public abstract class Framework {
    ...
    public abstract String [] listServices (); // list the available services
    ...
    public abstract void endAccess (); // releases resources used by the framework
    public abstract void destroy (); // disposes the framework
}
```

Figure 1. Framework abstract class.

This abstract class defines a neutral framework. As we can see, there is not any constructor. The constructor task will be done by next classes. They create the framework object: a `FrameworkAdapterFactory` (MI SDK) or a `Fwproxy` (EN SDK). This class only contains method definitions that apply on the created object.

It is necessary to define more abstract classes for all those classes that exist in each Java library. For instance, when the framework is successful accessed, next step is to obtain the service manager. In MI SDK is a service dependent class (`UserLocationAdapter`, `MessagingAdapter`, etc.) and in EN SDK is an `IpService`.

5.2 ISG framework class

When using MI SDK we actually want to use `FrameworkAdapter`.

We define a class that inherits from the abstract class all its methods, it implements them and adds the framework constructor. Figure 2 shows a possible implementation of the framework if MI SDK is used.

```
public class ISGFramework extends Framework {

    FrameworkAdapter fwISG; // ISG framework

    public ISGFramework() { // ISG constructor
        FrameworkAdapterFactory.createFrameworkAdapter("Sample","psw");
    }
    ...
    public String [] listServices() {
        return fwISG.listServices(); // list the available ISG services
    }
    ...
    public void endAccess () {
        fwISG.endAccess(); // releases resources used by the ISG framework
    }
    public void destroy () {
        fwISG.destroy(); // disposes the ISG framework
    }
}
```

Figure 2. ISG framework class.

As we can see, all general methods (abstract methods in framework class, Figure 1) use the particular methods defined in MI SDK. Most methods are very simple: they use MI SDK methods directly. For instance `destroy` method uses the `dispose` ISG method. Section 4.1.1 shows the MI SDK methods listed in Figure 2. Not all methods are so simple as shown, but to demonstrate the essence of this interface, this example is enough.

5.3 NRG framework class

```
public class NRGFramework extends Framework {

    FWproxy fwNRG; // NRG framework

    public NRGFramework () { // constructor
        Configuration.INSTANCE.load("configuration.ini");
        fwNRG = new FWproxy(Configuration.INSTANCE); // NRG constructor
    }
    ...
    public String [] listServices() {
        return fwNRG.listServiceTypes(); // list the available NRG services
    }
    ...
    public void endAccess (){
        fwNRG.endAccess(); // releases resources used by the NRG framework
    }
    public void destroy () {
        fwNRG.dispose(); // disposes the NRG framework
    }
}
```

Figure 3. NRG framework class.

When using EN SDK we actually have to use `Fwproxy`. As previous class, we define a class that inherit from the abstract class all its methods and adds the framework constructor. Figure 3 shows a possible implementation of the framework class if EN SDK is used.

5.4 The application

The application corresponds to the developed application. This sample is very simple and only creates the framework, then obtain an available service list, it shows them and finally release all sources. Figure 4 shows this simple application.

```
public static void main(String[] args) {

    Framework fw; // framework object
    String[] serviceList; // list of available services

    if (args[0].equals("NRG")) // discriminates the SDK
        fw = new NRGFramework(); // NRG framework creator
    else
        fw = new ISGFramework(); // ISG framework creator

    serviceList = fw.listServices(); // obtains the service list
    for (int i=0; i < serviceList.length; i++)
        System.out.println("Service =" + serviceList[i]); // shows the services
    fw.endAccess(); // releases resources used by the framework
    fw.destroy(); // disposes the framework
}
```

Figure 4. Sample application.

As we can see, only at the application beginning there is an explicit indication of which SDK we are using. Then, later the application doesn't care about that question. All methods used are those in the abstract class (Figure 1), but when the application runs, the real methods used are those in the ISG framework class (Figure 2) if the MI SDK is used or the NRG framework class (Figure 3) if the NE SDK is used.

6. CONCLUSIONS

The objective of P/OSA is to open an interface network to third parties, to permit developing new applications and services. We have shown that it is very powerful to develop new applications that use the telecommunication networks, although those interfaces shield most of the detail and eliminates most effort, those interfaces are very complex, difficult and hard to use. Several SDKs exist that solve these obstacles. They permit to use the Java language, simplifying the portability of the applications. This fact permits the use of very different terminals and resources, expanding, in a wide way, the use and reuse of the developed applications.

These SDKs have simulators that permit to develop application without accessing to a network. They can be done in a stand-alone way, without the difficulty that can add the fact to access a real network.

Those tools have a disadvantage: they don't support all interfaces defined in P/OSA, but they are enough for developing and test most applications.

We have shown the main steps that are necessary for developing an application: initialization, main functionality and finalization. We have applied those steps to a sample application to two SDKs: Lucent - MiLife ISG SDK and Ericsson NRG SDK. We have developed two sample applications, one for each SDK. They are only different when accessing to P/OSA functionality. So, it is necessary to develop an application for each SDK intended to use. To isolate the application from the particular SDK we propose to use an interface that hides the particular implementation of each SDK.

This interface is composed by a set of abstract classes that define a generic use of P/OSA and different implementations, depending on which SDKs want to be used. We have shown a partial implementation of this interface and we have applied to a sample application to show their structure and use. The whole interface is more complicated than that shown in this document, but this one is enough to show its philosophy and the way to develop applications.

This is only a preliminary work. We are currently working to expand these results: extend the interface to the whole SDK and to cover more SDKs. It is expected that few parts of SDKs will be more difficult to implement than others, and that few SDKs will be more complex than others. Due to this reason, we cannot assure that we will be able to apply this interface to all existing SDKs. We are currently working on these problems and we will report our results in the future.

ACKNOWLEDGEMENTS

This work has been partly supported by the Spanish administration (AgentWeb project, TIC 2002-01336) and is being developed within VISNET (IST-2003-506946, www.visnet-noe.org), a European Network of Excellence, funded under the European Commission IST FP6 program.

REFERENCES

1. Ericsson NRG Simulator and Software Development Kit (SDK) http://www.ericsson.com/mobilityworld/sub/open/technologies/parlay/tools/parlay_sdk
2. ETSI ES 202 915-1 V1.2.1 (2003-08) Open Service Access (OSA); Application Programming Interface (API) http://www.parlay.org/specs/index.asp
3. MiLife™ ISG SDK 3.0 Documentation. http://www.lucent.com/products/solution/0,,CTID+2019-STID+10490-SOID+966-LOCL+1,00.html
4. Ard-Jan Moerdijk, Lucas Klostermann. Opening the Networkswith Parlay/OSA: Standards and Aspects Behind the APIs. IEEE Network. May/June 2003.
5. Stephen M. Mueller. APIs and Protocols for Convergent Network Services. McGraw Hill. USA 2002.
6. Object Manager Group. http://www.omg.org
7. The Open API Solutions Application Test Suite http://www.openapisolutions.com/applicationtestsuite.html
8. J. Polo, J. Delgado. An easy way to develop mobile and wireless applications. 7th Int. Conf. Mobile and Wireless Communications Networks. To be published (Sep. 2005).
9. The 3rd Generation Partnership Project (3GPP) http://www.3gpp.org/
10. E. Vanem et al. Managing heterogeneous services and devices with the device unifying service. Implemented with Parlay APIs. 8th Int. Sym. Integrated Networks Manag., 2003.
11. E. Vanem et al. Realising Service Portability with the Device Unifying Service using Parlay API. International Conference on Communications, 2003.
12. J. Zuidweg. Next generation intelligent networks. Boston [etc.]: Artech House, cop. 2002.

IMPROVEMENT OF MPLS PERFORMANCE BY IMPLEMENTATION OF A MULTI-AGENT SYSTEM

Rana Rahim-Amoud, Leila Merghem-Boulahia, and Dominique Gaiti
ISTIT, University of Technology of Troyes, 12 rue Marie Curie, BP 2060, 10000 Troyes Cedex, France

Abstract: Multi-Protocol Label Switching (MPLS) is a network layer packet forwarding technology that provides flexible circuit switched traffic engineering solutions in packet switched networks by explicit path routing. However, the actual weakness of MPLS resides in its inability to provide application-level routing intelligence, which is a fundamental component especially for voice delivery. In this paper we propose to introduce a Multi-Agent System (MAS) within the MPLS network to improve its performance. The introduction of agents takes place into the decision points in MPLS at the flow level, and distributes traffic based on the quality of service required by the type of traffic. We also propose an intelligent framework for network as well as an architecture of our agent in order to improve the efficiency of the Quality of Service (QoS) within MPLS.

Keywords: MPLS; Quality of Service (QoS); Multi-Agent Systems (MAS); Artificial Intelligence (AI).

1. INTRODUCTION

The usage of the Internet has increased enormously in the last few years. At the same time, new real-time as well as non-real-time applications have emerged demanding for much improved Quality of Service (QoS) than the best effort currently proposed by the Internet. To adapt the network functioning to the requirements of these different types of applications, different networking solutions were proposed, such as the Integrated Services (IntServ)[1], its Resources reSerVation Protocol (RSVP)[2], and the Differentiated Services (DiffServ)[3]. Other proposals, like Active Networks

(AN)[4], introducing an adaptive management plane which allows real-time configuration management[5] and Multi-Protocol Label Switching[6], each within its objectives, have all succeeded in introducing amelioration to the global functioning of the network. They, however, still face miscellaneous limitations. To overcome these limitations, it became essential to find a single solution which makes it possible to exploit, to the maximum, the advantages of the existing solutions while playing the role of an essential complement. The evolution of the networks is marked by a tendency towards the intelligence and autonomy, removing any kind of centralization of the decisions, and opening the doors towards self-management and self-checking while ensuring the scalability, the adaptability and the survival of the networks. In line with this, recent research showed the effectiveness, the reliability and the robustness of the Multi-Agent System (MAS) for the dynamic management of complex and distributed systems. Our goal is to find the decision points in which we could introduce our intelligent agents, allowing as a result an improvement of the network performance and a satisfaction of users' requests. We chose for our study the Multi-Protocol Label Switching (MPLS) network which provides a set of services (Virtual Private Networks VPN and Traffic Engineering TE) in addition to the fast commutation that it allows between routers.

The paper is organized as follows. We first introduce MPLS protocol, its weakness and the current researches around this area. We then propose to find the decision points. In section 4, we propose an intelligent framework as well as an architecture of our agent. Finally we make a brief conclusion.

2. MPLS PROTOCOL

MPLS[6] is a new technology that associates labels with routers and uses these labels to forward packets by specifying the Forwarding Equivalence Class (FEC). FEC is a representation of a group of packets that share the same requirements for their transport. All packets in such a group receive the same treatment in the domain. MPLS domain contains two types of equipments LER (Label Edge Router) and LSR (Label Switch Router). The LERs are also called I-LSR (Ingress LSR) for the LSR that puts the label to an incoming packet and E-LSR (Egress LSR) to the one which removes the label from the outgoing packet to return it to its initial nature. LSR is a high-speed router device in the core of the MPLS network. In practice, the labels are distributed from the initialization of the network, related to the network configuration and routing tables which are established by a classic protocol like IGP. The path between 2 LERs is called LSP (Label Switched Path). As opposed to conventional IP forwarding, in MPLS, the assignment of a particular packet to a particular FEC is done just once, as the packet enters the network[7]. Many protocols can be used to distribute labels, LDP (Label Distribution Protocol)[8], CR-LDP (Constraint based–LDP)[9] that is an extension of LDP and RSVP-TE (RSVP with Traffic Engineering)[10]. LDP is a peer-to-peer protocol, while CR-LDP and RSVP-TE provide mechanisms for establishing end-to-end explicitly routed LSPs.

2.1 The weakness of MPLS

Currently the weakness of MPLS resides in its inability to provide application-level routing intelligence, which is a fundamental component especially for voice delivery[11]. Voice over IP (VoIP) is a critical application that requires intelligent routing alternation on the call level to prevent latency, delay, packet loss and jitter and this cannot be provided by MPLS. Adapting MPLS to VoIP traffic necessitates the distinction of different traffic types. Some solutions, which couple MPLS with DiffServ or RSVP, were proposed to solve MPLS limitations. According to[11], DiffServ over MPLS solution "*does nothing to solve the static route problem of MPLS*". If the QoS on a route degrades, MPLS plus DiffServ will not change the route, while it introduces complexity into the architecture. The implementation of RSVP with MPLS prevents an overbooking in the router from the start. However, RSVP "*is ineffective and impractical for solving the fluctuating demands of VoIP*"[11]. Even with these proposed solutions, MPLS remains unable to guarantee the QoS of incoming traffic, so it is very essential to find another solution.

2.2 Current researches

One of the critical research issues in MPLS is an efficient mapping of the traffic flows to LSPs. Such mapping provides an effective and efficient use of network resources and enables End-to-End QoS routing for real-time traffic. Having multiple LSPs for a destination is a typical setting which exists in an operational Internet Service Provider (ISP) network that implements MPLS technology. With multiple LSPs available for an egress node, the goal of the ingress node is to distribute the traffic across the LSPs by selecting the appropriate LSP from the available ones. Consequently, the network utilization as well as the network performance perceived by users are enhanced. According to this, Song et al.[12] discuss Load Distribution Management (LDM). Their main goal is to enhance the network utilization as well as the network performance by adaptively splitting traffic load among multiple paths. An LSP for an incoming traffic flow is dynamically selected based on both the current congestion level, and the length of the path in terms of the number of hops. LDM is intended for the best-effort traffic that does not impose any particular service requirement to the network. While LDM provides load-balancing solution and an efficient congestion control mechanism with increased utilization, it does not look into service differentiation for time-sensitive traffics like voice and video. Patek et al.[13] propose a simple alternate routing (SAR) in order to make a dynamic routing for aggregate traffic. SAR is based on three different parts: congestion discovery, selection of alternate paths, and allocation of traffic along alternate paths. Border nodes are responsible for satisfying all these three parts. The goal in SAR is to reroute traffic around congestion. SAR supposes that the differentiated services of the incoming traffic are already done and treats only the second phase. These above solutions remain unable to satisfy and to guarantee the QoS required for incoming traffic. We propose to introduce a MAS within the MPLS network to improve its performance and to guarantee a differentiation of services of incoming traffic. So we begin by finding the decision points.

3. DECISION POINTS

The first step of our research is to find the decision points. Once found, we can add our intelligent agents within these points.

3.1 The first decision point

Let us examine what occurs in the entry of the MPLS domain. Different types of traffic flows arrive to the entry of the MPLS domain. The classification of the packets is done just at the entry of the domain by I-LSR, by assigning a particular packet to a particular FEC. Within the domain, there is no reclassification, packets are just switched. Currently the most usable criteria to build FECs is based on the destination address or the prefix of the destination address, by taking advantage from the aggregation of flows that have the same destination. Aggregation may reduce the number of labels which are needed to handle a particular set of packets. It may also reduce the amount of needed label distribution control traffic. However, it does not take into account the type of traffic. Echchelh[14] has demonstrated by simulations with QNAP2 that the aggregation of different characteristics flows within the MPLS domain degrades the performances. Aggregation must be, consequently, based on the type of traffic and the required quality of service. In this paper we propose to introduce an intelligent agent on the level of each I-LSR router which is an efficient and a pertinent decision point. This agent will have as a role to examine the incoming flows and to create for each type of traffic a different FEC and consequently a different

LSP, even for traffics which have the same destination. As each E-LSR is at the same time an I-LSR for the packets forwarded in the opposite direction, then our intelligent agent will be introduced on the level of each LER within the MPLS domain, and consequently we obtain a MAS (Fig. 1). MPLS, in conjunction with path establishment protocols such as CR-LDP or RSVP-TE, makes it possible to set up a number of LSPs between a source-destination pair[6,9,15]. MPLS, with the efficient support of explicit routings enables assigning a particular traffic stream onto one of the available LSPs providing basic mechanism for facilitating traffic engineering.

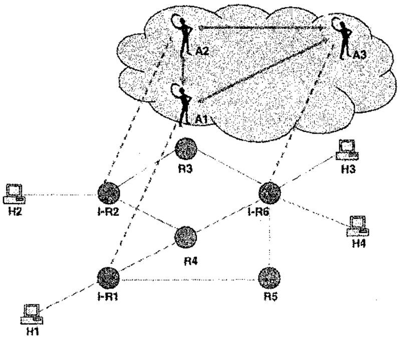


Figure 1. An intelligent agent is introduced on the level of each LER

The purpose of the introduction of our intelligent agent is to examine the current state of the network as well as incoming flows to, firstly select the suitable algorithm to start at the suitable moment and secondly to change the parameters of the selected algorithm.

3.2 The second decision point

According to[6], an LSR is capable of label merging if it can receive two packets from different incoming interfaces, and/or with different labels, and send both packets on the same outgoing interface with the same label. The MPLS architecture accommodates both merging and non-merging LSRs. In addition, MPLS contains procedures to ensure correct interoperation between them. To illustrate the importance of label merging, let us consider the example shown in Fig. 2. In the event that an LSR in the middle of the MPLS domain (R6) is merge-capable, the LSR simply acts as an Egress LSR to upstream neighbors and as Merge LSR to downstream neighbors[16].In the example (Fig. 2), if LSR R6 were merge-capable, it would perform aggregation of requests from upstream neighbors R1 and R2 thus reducing label consumption within the MPLS Network, the only condition is that the flows must be in the same FEC. Furthermore, label merging causes a small delay because it is designed for delay-insensitive traffic. Our proposal is to introduce an intelligent agent on the level of the LSR R6 having for role to activate the merge-capable of R6 when traffic is not delay-sensitive and to deactivate it in other cases. As a consequence the two LSPs will be aggregated into only one LSP from the point of intersection of the two LSPs.

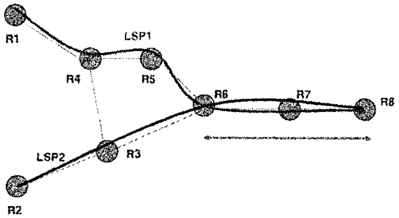


Figure 2. LSP attribution before the introduction of the intelligent agent.

4. AGENT AND MULTI-AGENT SYSTEMS

4.1 Definition

Agents and MAS are two innovative and interesting concepts for a great number of researchers in different domains such as simulation of road traffic[17-19], simulation of social phenomena[20,21], simulation of biological phenomena[22-24], medical simulation[25], etc. According to[26], an agent is a physical or virtual entity having trends and resources, able to perceive its environment, to act on it and to acquire a partial representation of it. It is also able to communicate with other peers and devices, and has a behavior that fits its objectives according to its knowledge and capabilities. The most important thing is that agents can learn, plan future tasks and are able to react and to change their behavior according to the changes in their environment. A MAS is a group of agents able to interact and cooperate in order to reach a specific objective. Agents are characterized by their properties that determine their capabilities. Different properties are defined like autonomy, proactive-ness, flexibility, adaptability, ability to collaborate and coordinate tasks and mobility. According to its role within its environment, the agent acquires some of these properties.

4.2 Agents and Multi-Agent System Architecture

Agents can be reactive, cognitive, hybrid or adaptive[24,27,28]. Reactive agents are suitable for situations where we need less treatment and faster responses (actions). Cognitive agents, on the other side, allow making decisions and planning based on deliberations taking into account the knowledge of the agent about itself and the others. Adaptive agents can adapt their actions and parameters to the changing situations. Hybrid agents are composed of several concurrent layers. Our approach is based on the architecture developed in[29]. In this architecture, two levels of agents are defined: the first one is composed of one agent called Master Agent which is a cognitive agent while the second one is composed of several reactive agents: LDP agent, Aggregation Agent, Queue Manager Agent, Routing Agent, Admission Controller Agent, etc (Fig. 3).

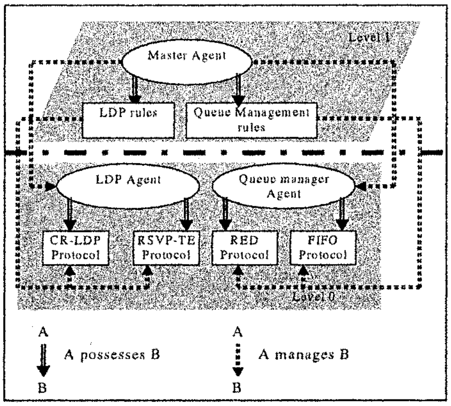


Figure 3. Two levels MAS architecture

The Master Agent observes the current router conditions and chooses the most appropriate protocols to the other agents under its responsibility. Each agent of level 0 has a set of protocols dedicated to the task it is in charge of. For example, LDP agent establishes some rules for selecting which label distribution protocol (LDP, CR-LDP, RSVP-TE...) to use under which circumstances. Deciding on what protocol is the most appropriate (and which must be adopted) will depend on the current QoS measures (loss, delay and jitter). In fact, there is no protocol giving optimal performance whatever the network condition. Hence, the need to adapt the current protocols to this new situation. Each Master Agent possesses a set of rules allowing to select the appropriate protocols to activate, and therefore to select the best actions to execute. These rules give the node the means to guarantee that the set of actions executed, at every moment, by its agents are coherent, in addition to be the most relevant to the current situation[30]. In order to minimize conflict situations, rules are organized in separate modules following the task they are interested (Fig. 3 shows an example).

4.3 Actions of the Master Agent

The actions of the Master Agent may consist in:

- letting the protocol running: this occurs when the protocol is still relevant to the current conditions;
- modifying the internal functioning of the protocol: this modification appears by updating the parameters on which the protocol depends;
- inhibiting the protocol: the inhibition happens when this protocol becomes useless regarding the current node's situation and rules;
- activating the protocol: the activation takes place if the Master Agent considers that this protocol is appropriate to the current node conditions. This activation may be accompanied by the inhibition of other protocols to avoid the coexistence of contradictory protocols.

The actions undertaken by the node have local consequences but may influence the decisions of the other nodes. In fact, by sending messages bringing new information on the sender node's state, a receiver's Master Agent rule may be triggered. This can involve a change within the receiver node (the inhibition or the activation of a protocol, etc.). This change may have repercussions on other nodes, and so forth until the entire network to be affected. This dynamic process aims to adapt the network to new conditions and takes advantage of the agents' abilities to alleviate the global system. We argue that these agents will achieve an optimal adaptive management process because of the following two points: (1) each agent holds different processes (protocols and adaptive selection of these protocols) allowing to take the most relevant decision at every moment; (2) the agents are implicitly cooperative in the sense that they possess rules that take account of the neighbors' state in the process of protocols' selection.

5. CONCLUSION

In this paper, we proposed to introduce a MAS within the MPLS domain in order to improve its performance. One of the step consisted of finding the decision points into MPLS which are especially identified on the entry of the domain on the I-LSR routers. The MAS is then situated into these decision points. The MAS has as role to set up multiple LSPs between an ingress-egress pair, and to distribute dynamically the incoming traffics to these LSPs. Traffics are distributed basing on their type and the required QoS. A basically two-layers architecture of MAS is also proposed in this paper. This architecture aims to dynamically select the appropriate protocol following the current QoS parameters. As future work, we intend to improve our architecture, to define our adaptation rules and to make a testbed.

References

1. R. Braden, D. Clark, and S. Shenker, Integrated Services in the Internet Architecture: an Overview, RFC1633 (1994).
2. J. Wroclawski, The Use of RSVP with IETF Integrated Services, RFC2210 (1997).
3. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, Architecture for Differentiated Services, RFC2475 (1998).
4. D. Tennenhouse, J. Smith, D. Sincoskie, D. Wetherall, and G. Minden, A Survey of Active Network Research, *IEEE Communications Magazine* **35**(1), 80-86 (1997).
5. M. De Castro, *Programmable and Adaptive Management of QoS in IP networks*, Ph.D thesis, (INT, France 2004).
6. E. Rosen, A. Viswanathan, and R. Callon, Multiprotocol Label Switching Architecture, RFC3031 (2001).
7. The international Engineering Consortium; http://www.iec.org.
8. L. Anderson, P. Doolan, N. Feldman, A. Fredette, and B. Thomas, LDP Specification, Network Working Group, RFC3036 (2001); http://www.ietf.org/rfc/rfc3036.txt.
9. J. Ash, M. Girish, E. Gray, B. Jamoussi, and G. Wright, Applicability Statement for CR-LDP, Network Working Group, RFC3036 (2002); http://www.ietf.org/rfc/rfc3036.txt.
10. D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivaan, and G. Swallow, RSVP-TE Extensions to RSVP for LSP Tunnels, Network Working Group, RFC3209 (2001); http://www.ietf.org/rfc/rfc3209.txt.
11. PRIMEDIA Business Magazines & Media Inc. (2003); http://infocus.telephonyonline.com/ar/telecom_questioning_mpls/.
12. J. H. Song, S. Kim, and M. J. Lee, Dynamic load distribution in MPLS networks, *Information Networking* **2662**, 989-999 (2003).
13. S. D. Patek, R. Venkateswaran, and J. Liebeherr, Simple alternate routing for differentiated services networks, *Computer Networks: the International Journal of Computer and Telecommunications Networking* **37**, 447-466 (2001).
14. Z. Echchelh, *QoS and Allocation of resources in ATM and MPLS networks,* Ph.D thesis (in French). (University of Bourgogne, France, 2001).

15. B. Jamoussi, L. Andersson, R. Dantu, L. Wu, P. Doolan, T. Worster, N. Feldman, A. Fredette, M. Girish, E. Gray, J. Heinanen, T. Kilty, and A. Malis, Constraint-Based LSP set up using LDP, RFC 3212 (2002).
16. A. Fredette, C. White, L. Andersson, and P. Doolan, Stream Aggregation, Work in progress (1997); draft-fredette-mpls-aggregation-00.txt.
17. S. El Hadouaj, A. Drogoul, and S. Espié, How to Combine Reactivity and Anticipation: the Case of Conflicts Resolution in Simulated Road Traffic, *MABS'2000 workshop* **1979**, 82-96 (Boston, USA, 2000).
18. A. Moukas, K. Chandrinos, and P. Maes: Trafficopter, A Distributed Collection System for Traffic Information, *CIA'98* **1435**, 34-43 (Paris, France, 1998).
19. A.L.C. Bazzan, J. Wahle, and F. Klügl, Agents in Traffic Modelling - From Reactive to Social Behaviour, *KI'99* **1701**, 303-307 (Bonn, Germany, 1999).
20. L. Ben Said, T. Bouron, and A. Drogoul, Agent-based interaction analysis of consumer behavior, *AAMAS' 2002, ACM,* 184-190 (Bologna, Italy, 2002).
21. R. Conte, N. Gilbert, and J.S. Sichman, MAS and Social Simulation: A Suitable Commitment, *MABS'98* **1534**, 1-9 (Paris, France, 1998).
22. A. Drogoul, When ants play chess, *From reaction to cognition* **957**, edited by C. Castelfranchi and J.P. Müller (Springer-Verlag, Berlin-Heidelberg, 1995), pp. 13-27.
23. A. Pave, F. Bousquet, C. Cambier, C. Mullon, P. Morand, and J. Quensiere, Simulating the Interaction between a Society and a Renewable Resource, *Journal of Biological Systems* **1,** 199-213 (1993).
24. J. Doran, Agent-Based Modelling of EcoSystems for Sustainable Resource Management, *3rd EASSS'01* **2086**, 383-403 (Prague, Czech Republic, 2001).
25. P. Ballet, F. Harrouet, and J. Tisseau, A Multi-Agent System to model a Human Secondary Immune Response. *IEEE International Conference on Systems, Man and Cybernetics (SMC'97),* 357-362 (Orlando, USA, 1997).
26. J. Ferber, *Multi-Agent System: An Introduction to Distributed Artificial Intelligence* (Harlow: Addison Wesley Longman, 1999).
27. S. Barber, and C. Martin, Dynamic Adaptive Autonomy in Multiagent Systems, Representation and Justification, *IJPR&AI* **15**(3), 405-433 (2001).
28. M. Wooldridge, Intelligent Agents, in: *Multiagent Systems: a Modern Approach to Distributed Artificial Intelligence,* (Weiss G. Press, 1999), pp. 27-77.
29. L. Merghem, and H. Lecarpentier, Agents: A Solution for Telecommunication Network Simulation, *Network Control and Engineering for QoS, Security and Mobility (NetCon'2002),* 165-176 (Kluwer Academic publishers, Paris, France, 2002).
30. L. Merghem, D. Gaïti, and G. Pujolle, On Using Agents in End to End Adaptive Monitoring, *E2EMon Workshop, in conjunction with MMNS'2003* **2839**, 422-435 (Belfast, Northern Ireland, 2003).

ENABLING MOBILE COMMERCE THROUGH LOCATION BASED SERVICES

Yufei Wu, Ji Li, Samuel Pierre
Ecole Polytechnique de Montreal, Department of Computer Engineering
5255 Decelles Building, Montreal
Canada
yufei.wu@polymtl.ca

Abstract The use of mobile telecommunications devices for commercial transactions, called mobile commerce (m-commerce), has been an emerging trend since the late 1990s. A killer application of m-commerce is Location Based Services (LBS). A host of new location-aware applications and services are emerging with significant implications for the future of m-commerce. The early stage infrastructure for enabling these services is just now reaching the commercialization stage. Strategic thinking in this area is rudimentary - there is not a clear understanding of issues associated with location services, such as business models. In this paper, we examine the technologies, applications, business models, and strategic issues associated with the commercialization of LBS, and give an outlook for future LBS development.

Keywords: Location Based Services, Mobile Commerce, Business Models

1. Introduction

Since the technological convergence of the Internet and mobile telecommunications networks in the 1990s, these technologies together have created the platform for a raft of mobile data services, including business-to-consumer (B2C) applications for financial services, gaming, email and news, and business-to-business (B2B) applications for tele-working, logistics, field sales and after-sales servicing. Worldwide, revenues from mobile (m-) commerce i.e., transactions over wireless telecommunications networks are expected to exceed 200 billion dollars by 2005.

A new type of m-commerce represents the "killer" application: applications that take the user's location into account in order to deliver a service (VanderMeer, 2003). Examples of such "location-based services" (LBS) include those that identify nearby locations, such as when a cellular telephone user seeking information about restaurants is provided only the set of choices in the immediate vicinity. In the next stage of e-commerce and m-commerce development, location-based services (LBS) are expected to play an increasingly important role in helping to differentiate one service provider from another (Van de Kar et al., 2001). For this reason, this paper provides an overview of this emerging class of mobile services, examining the LBS market potential, its technological bases, the potential services, the industry value chain and likely business models, significant policy issues, and potential future directions.

2. Location Based Services

A wide range of services that rely on users' location information have been conceived, although the markets are not yet mature. The main point is to remember that location is simply a useful bit of data that can be used to filter access to many types of geo-graphical information services (GIS). There are numerous ways to exploit location to provide more relevant information, or derive new services. It can be particularly powerful when combined with other user profile to offer personalized and location sensitive responses to customers (Searby, 2003). Van de Kar et al., (2001) distinguishes between emergency services, mobile operator services, and value-added services (VAS), focusing on the latter category as the primary e-commerce opportunity.

Table 1. Location based service applications

Location-Based Service	Application
Information/directory services	Dynamic yellow pages that automatically informs customer of location of nearest parking facilities, restaurants, etc. Travel, show, dinner reservations. Concierge services.
Tracking services	Tracking of children by parents. Locating lost pets. Locating friends in a geographic area. Tracking stolen cars. Tracking assets.
Emergency services	Roadside assistance. Search and rescue missions. Police and fire response. Emergency medical ambulance.
Navigation	Route description. Dynamic navigational guidance. Traffic status in the area.
Advertising promotions	Wireless coupon presentation, targeted ads, and promotional messages keyed to the location. Promotional alert when a sale of a desired product takes place. Customer identification in a store or a neighborhood.

In the VAS category, they describe a number of different service areas, including information, entertainment, communication, transaction, mobile office and business process support services. D'Roza et al., (2003) classify services into two broad categories: those that are requested by users once their location is determined, and those that are triggered automatically once a certain condition is met (e.g. a boundary is crossed). We might consider the former set to be "pull" services and the latter to "push" services. In addition, they also identify five groups of application areas: communication, fleet management, routing, safety and security, and entertainment. We can also classify services according to whether they apply to consumers, business customers, or employees in a firm. Some of the most commonly discussed services are briefly described in Table 1.

3. Location Positioning Technologies

One or more location methods can be used to determine the position of user equipment for LBS. It is also possible to distinguish between methods that are most useful inside and outside buildings. Leading candidates for indoor location identification include short-range radio, such as Bluetooth, and infrared (IR) sensors (Barnes, 2002). For example, developers could use Bluetooth or IR to build an automatic tour-guide system, such as for an art gallery; as the tourist with a suitably enabled PDA device moves into range of a piece of artwork, it could send out a signal that automatically displays information related to the artwork on the screen (Tseng et al., 2001). However, interesting though this is, the focus is on roaming, location-aware technology used largely outside buildings. For a detailed examination of the benefits and applications of short-range wireless technologies, see (Barnes, 2002).

Location techniques operate in two steps-signal measurements and location estimate computation based on the measurements which may be carried out by the user equipment or the telecommunications network. Subsequently, positioning techniques can be categorized into several varieties, each with its advantages and disadvantages. The main types are cell-location, advanced network-based, and satellite-based positioning. Three of the main categories of positioning methods are shown in Table 2, in order of increasing accuracy.

4. LBS Value Chain and Business Models

There are different players that may be involved in bringing location-based services to the market. Among the parties involved are:

- Geographic information service (GIS) and other content providers who offer a range of mapping services and geographically oriented content, often accessed via a server known as a geoserver.

- Service providers who aggregate GIS and other content to create services.

Table 2. Location based service applications

Type	Methodology	Advantages	Disadvantages
COO	Network-based technology. Base stations use RF signals to track mobile devices.	Relative widespread infrastructure No handset modification required Rapid implementation and low cost.	Hard to pin down user's extract location to a few meters
GPS	Handset-based technology. 24 low orbit satellite network. Triangulation used to determine exact location.	Outdoor precision within 5 meter range. No dependent on the network.	Expensive. User device must be in direct line of sight. Device needs special embedded chips
WLP	Positioning methods by using UWB, RFID, WLAN etc.	Positioning precision could reach less than 1 meter. No need for expensive network infrastructure. No need using mobile operators' resources, such as frequency.	

- Application vendors who package services for mobile operators.
- Location middleware providers who provide tools to facilitate mobile operators' use of various applications from different providers.
- Mobile operators who manage the infrastructure, collect the position data, offer the service to the end subscribers, and perform billing and collection services.
- Location infrastructure providers who sell the mobile location centers and other hardware and software to network operators.
- Handset manufacturers who sell devices capable of interacting with LBS.

Each of these parties stands to earn revenue from location based services, but the whole value chain requires standard data formats and interfaces to work effectively (Spinney, 2003). If each individual application has its own proprietary format, the costs to launch a suite of services for consumers would be prohibitive for mobile operators.

The business models for LBS will most likely vary considerably across services. Sources of revenue for service providers may include subscription fees for a bundle of service available via a portal, subscription fees for specific services, advertising, connection and airtime fees, fees for content, transaction fees or margins on the price of products ordered (D'Roza et al., 2003; Sadeh, 2002). In some cases, such as for emergency 911 services, the operators may collect revenue to pay for the services through regular phone subscription fees. Another source of revenue may come from businesses that pay a fee in order to be included in location-based business directories, even if the service does not include any push-based advertising. Indeed, many privacy advocates have expressed opposition to the use of advertising that is pushed to the client, rather than specifically requested, suggesting that this is unlikely to be a viable revenue stream.

Most likely, LBS will use various combinations of revenue models. For example, customers may be offered the choice between advertiser and non-advertiser supported services, with the former provided at no cost and the latter provided for a fee.

In addition, many location-based services will be offered as a business service to companies, targeting their employees. In these cases, the service will resemble something like a private network, with bulk or volume discounts offered to large business clients. Individual employees will not be charged. For firms, the motivation will be to enhance employee productivity and make particular business processes more efficient. Some analysts, in fact, believe that this will be the primary early market for location-based services.

5. Commercialization Issues

Mobile commercialization of location based services to increased revenues and profitability depends on how the different players in the value chain resolve the key strategic commercialization issues they are facing. These issues include the selection of the underlying location-aware technology, ownership of the location data, interoperability, and mitigating privacy concerns. We briefly highlight these issues in this section.

Privacy

Privacy handling is a major issue in LBS deployment and provision and a critical success factor to the wide acceptance of this technology framework. The terms privacy handling consolidate issues like ownership of location information, use of location information, disclosure to service providers, etc. Skepticism arises as to where and how privacy handling should take place within the LBS provision chain. Existing proposals from operators and standardization bodies specify a priority scheme whereby the core network elements (e.g., Home Location Registers) have master control on location information. The provision/disclosure of such information to other entities (e.g., location servers, LBS serving nodes, ASPs) is subject to subscriber needs (e.g., registration information) and regulatory frameworks.

Economic Control of Location Information

In the value chain for the provision of location-based services, depending upon which method of determining location is used, service providers may be dependent upon cellular network operators for access to customers' location data. If the network operator had a competing location-based service, then they may have an incentive to either not make this information available, or to make it available on terms that place the competing service provider at a disadvantage. Policy makers will need to make clear exactly what the obligations are for the provision of location data, in addition to ensuring that I formed consent is enforced.

Analysts have also cautioned network operators to avoid the "walled garden" approach to location-based service provision. Operators might be lured by the opportunities for a larger share of the revenue if they provide their own restricted and branded set of services to users. Experience with WAP portals, and earlier generations of information services suggest that this strategy will fail. On the other hand, the fastest growth of wireless data services appears to be in Japan's iMode system, which does not restrict customers' access to third party services that are independent of the operator's brands (FCC, 2001). I-mode also offers a full complement of location-based services known as i-area (Spinney, 2003).

Quality of Service

Operators have chosen different methods for determining location, and with varying costs and accuracy. Some location-based services may require more accuracy than others (Adusei et al., 2004). For example, driving directions may require an accuracy of 30 meters, while location-sensitive billing or mobile yellow pages may only need to locate a user within a range of 250 meters (Spinney, 2003). Moreover, if operators are using a GPS solution that requires a minute or more for the time to first fix, then such delays might result in quite inaccurate positioning in fast moving vehicles. Customers may not be able to obtain the requisite quality of service on a particular provider's network.

A more serious quality of service issue faces service providers who use the unlicensed spectrum. The introduction of wireless LANs in public settings, with fee-based access, creates an expectation for a certain quality of service. However, service providers might have little control over others' use of the same spectrum in that area, since it is unregulated and services might suffer from interference.

Another related issue is the extent to which location-based services will interoperate with different user terminal equipment. If a user roams, for example, to another state, region, or country, will their terminal equipment still be able to work with the available network infrastructures to determine location and provide LBS? Manufacturers and operators are working together in the Location Interoperability Forum to help avoid fragmented supply of services.

Interoperability

We have seen previously that there is an almost bewildering variety of technologies, devices, networks and location-sensors, upon which location-based experiences might be developed. Our third major technical challenge is therefore interoperability. Interoperability of networks is a key issue, including roaming between different network providers and eventually exploiting multiple network technologies and/or architectures within a single experience, for example, simultaneously using cellular telephony to communicate with remote servers and peer-to-peer Bluetooth connections in order to communicate with other users nearby. Similarly, it makes sense to combine multiple location-sensing technologies so that they reinforce one another, an approach known as sensor fusion. Given that such heterogeneity is likely to be a feature of location-based experiences, it is important to develop suitably flexible middleware to support application developers in a 'pick-and-mix' approach to combining devices, networks and sensors.

6. Future Directions

It may seem a bit premature to discuss the future of the location-based service industry given its relative state of immaturity. Nonetheless, the extensive work in the computer science community on pervasive and context-aware computing further suggests that future systems will incorporate more than location information and data drawn from personal profiles in the provision of services. Rather, embedded sensors are likely to enrich the services with a wide range of additional context data. Additionally, the proliferation of unlicensed wireless and the rapidity with which both wireline and cellular operators have moved to integrate 802.11b options into their portfolio of services suggests that convergence across between indoor and outdoor systems is likely to occur.

Convergence Between Location Sensing Technologies

As WiFi systems proliferate, it is possible that they may supply many LBS simply by virtue of being able to assume that connected parties are within range of a particular base station. This may threaten the viability of some services offered by mobile operators, since increasingly, WiFi hotspots are either free or very low in cost. On the other hand, seamless provision of location and context-aware services require a mix of technologies (Spinney, 2003; Unni et al., 2003). A consumer may initiate a request from his or her car for all businesses in the local area using a GPS equipped PDA or cellular phone. The service provider may provide navigational services to direct the consumer to the appropriate location. Upon entering the business, a local WiFi network may provide additional information, and guide the consumer to their desired product. Some method for handover of such applications is needed, without requiring consumer to re-input product preferences. Spinney (2003) discusses handover methods that rely on both the location of the mobile phone user and the location of the indoor "hotspot." He further sees future handsets incorporating both cellular and 802.11 capabilities. These connections need to be seamless and without effort, especially if users are paying for access to services.

Many applications lose their value if customers, or business users are out of reach once they enter indoor environments. For this reason, we may see greater efforts to integrate applications across the variety of technologies for location-based services.

Context Awareness

Up to now, most service providers are focusing their attention on location as the primary type of information to use when customizing services for subscribers. However, researchers active in the area of mobile computing consider location to be only one aspect of a users' context. Over the past decade, computer scientists have been exploring a variety of ways to make computer-based applications sensitive to location as well as other contextual information (Hightower et al., 2001; Schmidt et al., 1999). Context may include both user provided profile information, as well as other aspects of context that may be detected by the system.

Some of the research on context-aware computing has quite direct implications for e-commerce. First, much of the research has been completed in indoor environments, using such location and context detection technologies as infrared, ultrasound, and low power radio (Schmidt et al., 1999). Hence, it has the potential to fill in an important gap in the coverage afforded by GPS and some public cellular network-based location services. Unlike straight wireless LANs, which generally do not determine the location within a building or room, these systems do provide precise positioning indoors. Retailers, for example, may be interested in helping shoppers find products once they are already inside malls or stores, and providing highly local navigation aides (Schmidt et al., 1999). Depending upon the granularity of the position detection, as well as user preference information, changing information could be provided to PDAs as shoppers moved about a store or mall. Such systems have obvious application in museums and other tourist areas.

Other types of context information may come from sensors deployed on machines. Automotive firms talk about "telematic services" as including data about the state of particular components on vehicles, such as their need for repair. This information can be sent from vehicles to car dealers, setting up preventative maintenance appointments prior to breakdowns.

7. Concluding Remarks

Our overview of location-based services reveals that the market potential is thought to be significant, driven in part by the deployment of automatic location identification systems for emergency response. There are, however, significant barriers to overcome. Technological barriers result from the diversity and cost of approaches to location determination, creating a complex set of choices for operators and potential interoperability problems that, if unsolved, are likely to stifle development. There are many exciting services under development, and some have been operating successfully in such markets as Japan for several years. Innovative applications such as location-based games have achieved a following in Sweden and been introduced into other markets. Despite the promise of LBS for consumers, however, privacy concerns, quality of service problems, fair access to location information, and the lack of standards for technology and service providers may hinder market development and represent critical policy issues to be resolved. Finally, in the area of potential future directions, it is evident that location is merely a starting point for personalization and context-aware services that use other relevant information when constructing service offers. Moreover, the rapid deployment of alternative wireless technologies, such as Wireless Fidelity (WiFi or 802.11) is both a threat and an opportunity for cellular operators, and will likely shape the future development of LBS.

References

Adusei, I. K, Kyamakya, K., Erbas, F., 2004, Location-based services : Advances and Challenges. *CCECE.*

Barnes, S.J., 2002, Under the skin: Short-range embedded wireless technologies, *International Journal of Information Management*, **22**(3): 165-179.

D'Roza, T., Bilchev, G., 2003, An overview of location based services, *BT Technology Journal* **21**(1): 20-27.

Hightower, J., Borriello, G., 2001, Location systems for ubiquitous computing, *IEEE Computer*, **8**: 57-66

Sadeh, H, 2002, *M-Commerce: Technologies, Services, and Business Models*, Wiley, New York.

Schmidt,A., Beigl, M., Gellersen, H., 1999, There is more to context than location, *Computers and Graphics Journal* **23**(6): 893-902

Searby, J., 2003, Personalization- An overview of its use and potential., *BT Technology Journal* **21**(1): 13-19.

Spinney, J., 2003, A brief history of LBS and how OpenLS fits into the new value chain. *Java Location Services Newsletter* .

Tseng, Y.C., Wu, S. L., Liao, W. H., Chao, C. M., 2001, Location awareness in ad hoc wireless mobile networks, *IEEE Computer* **34**(6): 46-52.

Unni, R., Harmon, R., 2003, Location-based services:Models for strategy development in mM-Commerce. *Proceedings of PICMET-2003, Portland International Conference on Management of Engineering and Technology*, Portland, OR.

Van de Kar, E., Vetter, R. (October 2001), The development of location based mobile services, *Edispuut Conference*, Amsterdam.

VanderMeer J., (March/April, 2003), What is the difference between M-Commerce and L-Commerce, *Business Geographics*.

INTER-DOMAIN TRAFFIC ENGINEERING USING MPLS

Meral Shirazipour[1], Samuel Pierre[1], Yves Lemieux[2]

[1] Mobile Computing and Networking Research Laboratory (LARIM),
Department of Computer Engineering, École Polytechnique de Montréal,
C.P. 6079, succ. Centre-Ville,.Montréal, Qué., Canada, H3C-3A7
{meral.shirazipour, samuel.pierre}@polymtl.ca
http://www.larim.polymtl.ca/
[2] Ericsson Research Canada
8400 Decarie Blvd., Town of Mount Royal, Québec, Canada, H4P-2N2
{yves.lemieux}@ericsson.ca

Abstract. In the Internet, the traffic crosses between two to eight autonomous systems before reaching its destination. Consequently, end-to-end quality of service requires provisioning across more than one domain. This paper proposes a new scheme for introducing MPLS technology into an inter-domain environment. Results obtained using the OPNET simulation platform show that extending MPLS across AS boundaries can improve the QoS perceived by the end users. This means that inter-domain traffic engineering is a promising solution for a QoS aware Internet.

1 Introduction

For economic reasons, the Internet backbone is the transport network chosen by current and future generation service providers. The foreseen applications have different quality of service (QoS) requirements, in particular regarding availability, delay, jitter, packet loss, etc. It is the job of network operators to assure an adequate support for these applications. They will need to guarantee various levels of QoS to dissimilar traffic flows, while maintaining their own profitability. Numerous architectures have been proposed, or even implemented, in order to increase revenues by maximizing network utilization. But, providing a guaranteed QoS while preserving earnings is a complicated task and remains an open issue.

When speaking of guaranteed QoS, the most important concern is the QoS perceived by the end users. This means that the QoS and application requirements listed above need to be sustained end-to-end, from one user terminal to the other user terminal. It is also known that traffic usually traverses more than one domain in the Internet before reaching its destination (between two to eight domains) [8]. Therefore inter-domain traffic engineering mechanisms are essential in order to achieve end-to-end QoS guaranties.

Providing end-to-end QoS is one of the most important challenges in the Internet. The Internet is composed of about 13000 distinct domains [3], each belonging to a different company or ISP. These domains are under different administrations and are called autonomous systems (AS). An AS is a set of routers functioning under the same administration.

The inter-domain traffic engineering difficulty in the current Internet architecture is caused by the various QoS policies enforced with often a different definition or implementation from one domain to the other. Moreover, topology and link state information is essential for effective inter-domain traffic engineering; but for scalability and privacy reasons the Border Gateway Protocol (BGP) does not propagate this information.

The literature proposes different methods for performing inter-domain traffic engineering [3], [4], [2], [11], [7], [9]. Some of these techniques, e.g. BGP traffic engineering, are already in use in the Internet. Others, such as inter-domain LSP (Label Switched Path) setup or community attribute extensions, are pending proposals at the IETF. BGP traffic engineering is performed by tuning route advertisements. Such tuning mechanisms have their limitations. They are trial and error based, give little control over the end-to-end path taken, lack optimality and have no notion of QoS. The other proposed technique, inter-domain MPLS, is more useful in controlling the traffic, but is not fully defined. It still does not specify how to maintain an end-to-end control over the traffic. This work consists in the description of a method for deploying end-to-end LSPs based on already proposed extensions to RSVP-TE for a similar purpose [9]. In this work, we study the usefulness of end-to-end LSPs by means of simulation results in OPNET. Section 2 gives a brief review on related works regarding the RSVP-TE extensions. Section 3 introduces the proposed mechanism for end-to-end inter-domain LSP setup. Section 4 presents the simulation model and results. Finally, section 5 concludes by presenting related future works.

2 Background and Related Work

Multi-Protocol Label Switching (MPLS) [10] is a packet forwarding framework that performs label switching between layer 2 and layer 3 protocols in the OSI model. The original benefit of MPLS was faster packet forwarding, which nowadays is achievable by more advanced hardware. These days MPLS is mostly used for traffic engineering purposes, to deliver QoS and differentiated services, to offer Fast-Reroute resiliency mechanisms, and to support virtual private networks (VPN). If the MPLS framework is extended beyond a single domain, the technology can be very useful for inter-domain traffic engineering and end-to-end QoS provisioning. The inter-domain extension of MPLS essentially involves the signaling protocol used for the exchange of information between MPLS nodes during LSP setup. Many proposals are made in the literature regarding inter-domain MPLS. Inter-domain LSP setup using bandwidth management points is proposed by [7], but because of its revolutionary nature it is not a pragmatic method in today's Internet. A practicable traffic engineering mechanism designed for the current Internet should allow a smooth migration of the existing technologies towards the proposed one. This can be achieved more easily if the proposed method is an extension to already operating mechanisms. The inter-domain MPLS proposal with RSVP-TE extensions [9] is a method that should consequently be considered. The choice of RSVP-TE in this work can be explained by the fact that it is the most popular and mainly implemented signaling protocol for intra-domain MPLS deployment.

2.1 Intra-domain MPLS versus Inter-domain MPLS

In a MPLS network, Fig. 1, the packet is only routed once at the ingress LER (Label Edge Router) where it receives a label. It is then forwarded through the network following the LSP assigned to its label. At each LSR (Label Switched Router) the label is swapped with another label of local significance (local to the node). When the packet emerges at the egress LER, the last label is removed and the packet is forwarded to its destination with normal IP, or towards its final destination via another network. In each node, packets assigned to a given label belong to the same FEC (Forwarding Equivalence Class). A FEC is a logical entity that designates a group of packets undergoing equivalent forwarding in a given node. During normal IP operation, for each possible next hop, a router usually creates a different FEC. With MPLS, other more advanced criteria can be used to designate a FEC. Criteria such as source-destination address pairs and destination address-ToS pairs are such examples, leaving lots of flexibility for traffic engineering in MPLS networks.

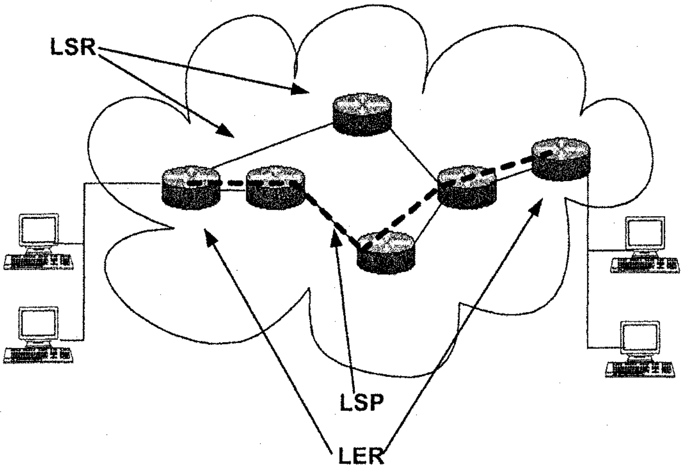


Fig.1: MPLS network

Several requirements for inter-domain MPLS deployment are discussed in [12]. First there is the service providers' need to keep internal resources and intra-domain LSP paths confidential. This implies that global topology information is not available for inter-domain LSP setup. Another requirement for the proposed inter-domain

MPLS mechanism is scalability, in terms of IGP flooding, BGP message exchanges, and signaling extensions. Nonetheless, the usual goals of intra-domain MPLS traffic engineering must be met. These fundamental MPLS goals are

- Support of end-to-end quality of service mechanisms
- Optimization of network resources
- Fast failure recovery methods

2.2 RSVP-TE

In RSVP-TE [1], RSVP [5] is enhanced to enable routers supporting both RSVP and MPLS to associate labels with RSVP flows. To support MPLS, RSVP-TE introduces new objects that will be carried inside RSVP *Path* and *Resv* messages. The LABEL_REQUEST object is carried inside a *Path* message initiated by the LSP's ingress LER. Its purpose is to request the egress LER to initiate a reservation and establish an LSP along the path followed by the *Path* message. The egress LER assigns a label to the LSP that is being created. It then puts that label in the LABEL object of a *Resv* message and sends it to the next node upstream. At each node, a local label is assigned to the LSP, the LABEL object is updated and sent to the next node upstream. This procedure ends at the ingress LER, the LSP being created in this way. RSVP-TE also introduces two other important objects for traffic engineering purposes: the Explicit Route Object (ERO) and the Record Route Object (RRO). These objects are used to allow the LSP to be established along a predefined route rather than the one determined by the IP routing protocol. The predefined route can be calculated by different means, e.g. using manual configuration or by a PCE[1]. The explicitly routed LSP could be used to avoid congested routes, to take disjoint routes during fault recovery mechanisms, or simply to offer the required QoS.

2.3 Extensions to RSVP-TE

In [9], the authors extend the use of RSVP-TE for the deployment of inter-domain LSPs. The goal of the authors is to provide recovery mechanisms for inter-domain link failures, but their method can be refined for the purpose of end-to-end LSP deployment and QoS provisioning. They propose extensions to RSVP-TE that fulfill both the confidentiality and LSP protection requirements. In addition, their method does not disturb the already in place inter-domain routing and signaling protocols. Their LSP establishment method is discussed in the following paragraphs.

For the establishment of intra-domain LSPs, the LER that sets up the LSP tunnel has topology information obtained from the IGP protocols. However, for inter-domain LSPs, the source of the LSP does not have detailed inter-domain topology information. The only information the source router has is on its own domain, obtained by its IGP, along with the route information distributed by BGP. Hence, the

[1] The Path Computation Element concept is being defined at the IETF by the PCE Working Group

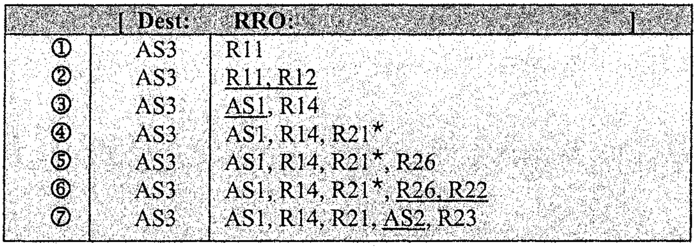
source LSR or PCE cannot determine a complete path for the LSP to the destination AS. Moreover, prior to the establishment of the LSP tunnel, the ingress router does not know the IP address of the remote egress router. At this point, the only information the ingress router has is the destination's address prefix and AS number.

To answer this problem, [9] proposes the establishment of inter-domain LSPs based on a destination prefix, or on an AS number and a prefix. For the first case, the LSP is created by forwarding a *Path* message through the network until reaching an LSR with an IP address that matches the prefix. The second case consists again in forwarding a *Path* message on the basis of the destination prefix until reaching an LSR that is part of the specified AS.

It should be noted that the prefix or AS and prefix information are necessary to send the first *Path* message. However upon the reception of the first reservation message, the egress IP address is obtained. It is possible to use this IP address to establish future LSPs to that destination, for backup or load balancing purposes.

Fig. 2 shows an example of the further extension to RSVP-TE that answers the confidentiality requirement of inter-domain MPLS deployment. Here LER R11 of AS1 wants to create an LSP towards AS3. It sends a *Path* message towards that destination but also needs to record in the RRO object the route followed by that LSP. However, recording the complete path of the LSP violates the confidentiality requirement of each AS to keep its internal routing information private. The proposed method [9] consists in aggregating the RRO object in such a way that the only information an AS will divulge about itself is its entry router, its number and its exit router. Table 1 explains Fig. 2 by showing the contents of the RRO object at each point along the LSP. Point 4 demonstrates how the LER R21 of AS 2 is marked as the entry point. Point 7 shows how the exit point of AS2, R23, performs RRO aggregation and hides all the information back to the entry point R21. The source LER, R11, can send subsequent *Path* refresh messages with an ERO containing the path recorded in the RRO. At the entrance of each AS, the ERO will be updated with the RRO of that path-state, containing the information confidential to the AS, which was recorded in the entry LER.

Table 1: *Path* message at different instances in Fig.1



	[Dest:	RRO:]
①	AS3	R11
②	AS3	R11, R12
③	AS3	AS1, R14
④	AS3	AS1, R14, R21*
⑤	AS3	AS1, R14, R21*, R26
⑥	AS3	AS1, R14, R21*, R26, R22
⑦	AS3	AS1, R14, R21, AS2, R23

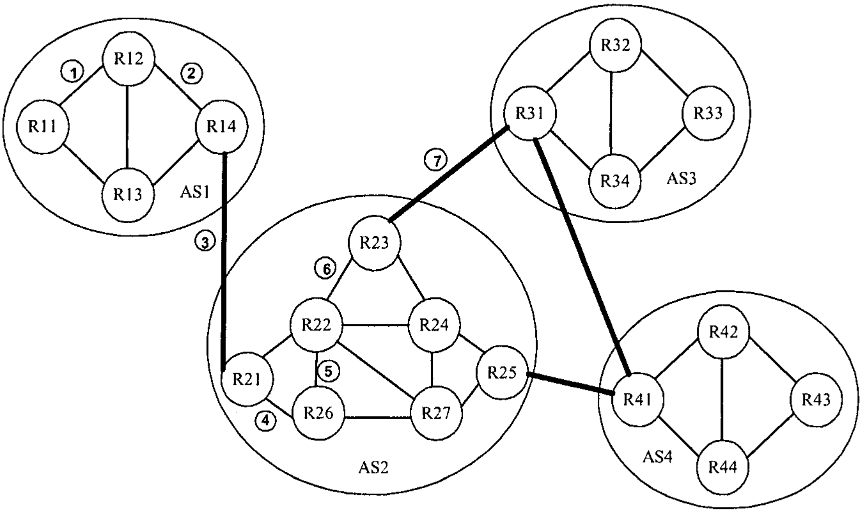


Fig.2: RRO aggregation

3 Proposed Mechanism for End-to-end LSP Setup

To provide an actual end-to-end QoS, the access network which connects the user terminal to the Internet must also be considered. But due to the heterogeneity of access technologies our contribution covers the end-to-end path, up to the last AS, excluding the access network itself. The reader should note that an actual end-to-end mechanism should also cover the access network of the user terminal. Nevertheless, we suggest extending the proposed end-to-end LSP method to the IP operating access networks, in order to support the traffics considered.

To assure the QoS in an inter-domain environment, we use LSPs that cover the end-to-end path of the traffic. The path taken by the LSPs can be optimized using LSP optimization techniques present in the literature [6]. For end-to-end LSPs crossing many domains, this optimization can be performed in a distributed fashion by each AS or by designated PCEs.

For establishing inter-domain LSPs, we make use of the already explained extensions to RSVP-TE [9]. In our suggested mechanism, the head end LER would first give an end-to-end label to the LSP (from the global-LSP subset of its label space), on top of which it would place another label (from the local-LSP subset of its label space). As shown by Fig. 3, at each AS, the packet would be forwarded with a label of minimum depth two [8]. The label at depth 1 will be used to identify the packet as being on an inter-domain LSP. The label at depth 2 will be the actual inter-domain label. In doing so, we intend to propose a differentiation inside the label space used by packets following only the traditional intra-domain LSP and packets following an inter-domain LSP. This differentiation, or packet classification, shall be useful for future MPLS based inter-domain traffic engineering techniques such as protection, recovery, security, etc.

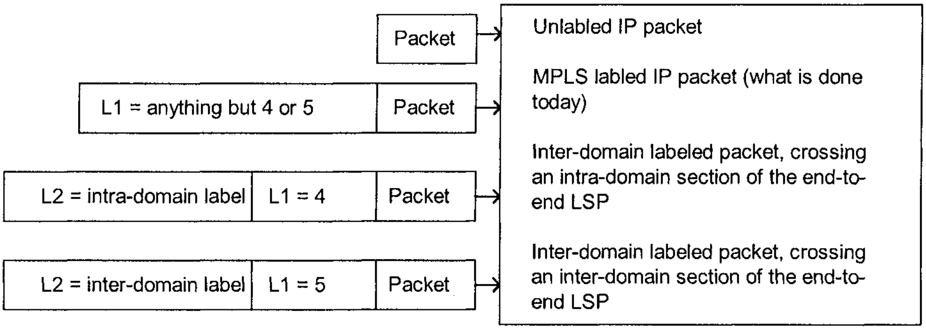

Packet
Unlabled IP packet
L1 = anything but 4 or 5
Packet
MPLS labled IP packet (what is done today)
L2 = intra-domain label
L1 = 4
Packet
Inter-domain labeled packet, crossing an intra-domain section of the end-to-end LSP
L2 = inter-domain label
L1 = 5
Packet
Inter-domain labeled packet, crossing an inter-domain section of the end-to-end LSP

Fig.3: Inter-domain labels (Level 1 and Level 2 labels)

Upon receiving an unlabeled packet, the ingress LER uses the FEC to forward that packet. This decision consist in forwarding with plain IP, i.e. no QoS; along an intra-domain LSP, i.e. leaving part of QoS decisions to further ASs encountered along the way; or through an end-to-end *inter*-domain LSP, i.e. providing end-to-end QoS. If the ingress LER decides to label the packet, it will use the FTN[1] to map the FEC to an NHLFE[2]. Using the information in this NHLFE, it will perform forwarding decisions on the packet. If the incoming packet is already labeled, the ILM[3] will be consulted to forward the packet. In both cases, the labeling will be performed as follows:

- In the case of an LER that does not support inter-domain MPLS, the labeling will be done as described in [10].
- In the other case, the label at level two will be the actual value of the label, while the label at level one will be set to:
  - ✓ 4, if the packet is on the intra-domain part of an inter-domain LSP
  - ✓ 5, if the packet is on the inter-domain part of an inter-domain LSP

4 Simulation Model and Preliminary Results

Our simulation model consists in the implementation of inter-domain LSP scenarios in OPNET modeler 10.5. Fig. 4 shows our network model. This network consists of four ASs. The host applications are located in Montréal, connected to AS 1, and Washington D.C., connected to AS 3. AS 1 can transmit to AS 3 through the path AS 1→AS 2 →AS 3 or through the path AS 1→AS 2→AS 4→AS 3. IGP/BGP routing protocols would normally favor the shortest path, that is AS 1→AS 2 →AS 3. If the

[1] FEC-to-NHLFE, unlabeled packet mapped to an NHLFE [10]

[2] Next Hop Label Forwarding Entry, contains packet's next hop and operations to be performed on label or on packet [10]

[3] Incoming Label MAP, maps incoming labels to NHLFE [10]

AS 2 connection to AS 3 becomes congested or cannot sustain the required QoS anymore, BGP will not reroute the traffic through the AS 2→AS 4→AS 3 path. We demonstrate that by using an inter-domain LSP the traffic can be forwarded through the desired inter-domain route and thus take the less congested one. Moreover, in case of an inter-domain link failure, the LSP fast recovery technique improves the delay experienced by the traffic compared to the time taken by conventional BGP routing to recover from the failure.

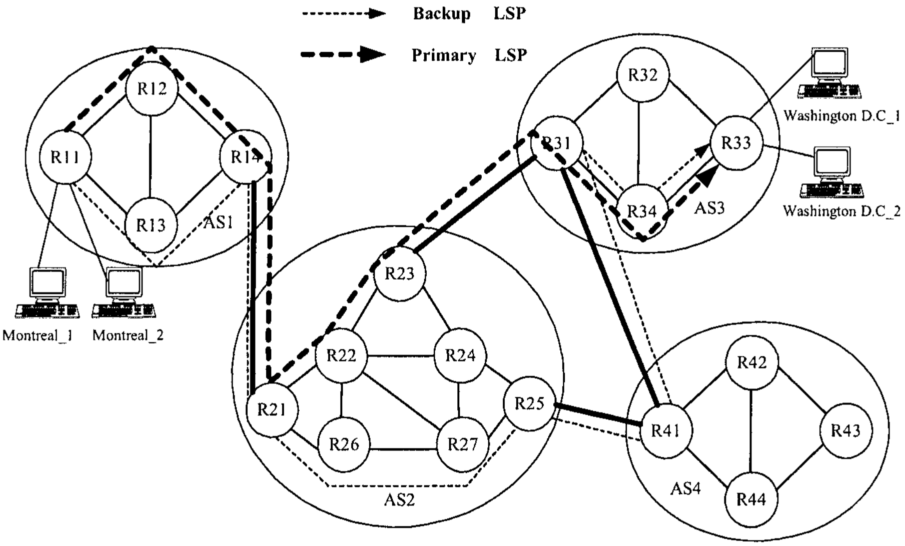


Fig.4: Network model: Internet backbones (ASs)

As depicted in Fig. 4, two LSPs join the source and destination ASs. The following simulation scenarios are sufficient to prove the effectiveness of our scheme compared to normal BGP routing:

Scenario 1: Link R23-R31 becomes congested
Scenario 2: Link R23-R31 fails
Scenario 3: Router R23 fails

In each of the above cases, the traffic is initially forwarded along the R23-R31 link, and after the event of a heavy congestion or a failure, it is switched on the backup LSP in order to avoid the problematic link or node. The simulation results consisted in measuring the QoS gain brought by the use of inter-domain MPLS compared to normal IP-BGP routing. The QoS parameters of interest were the mean delay and the mean jitter experienced by the traffic. Two types of traffic were used: voice traffic and video conferencing traffic. Fig.5 and Fig. 6 show that in the cases of link or node failure, the mean delay and mean jitter experienced were improved for both types of traffic. This is with the exception of the mean jitter experienced by the video conferencing application in the event of a node failure. The difference is insignificant due to the relatively small values of the jitter. The actual difference between the mean jitter with our scheme compared to BGP routing is 0.00004 s. In the case of congestion avoidance, it is seen that the gain is negative for both types of traffic and for both QoS parameters. The reason for that is that the congestion level of the link was not acute. Since our backup LSP takes a longer route to the destination, the delay and jitter experience with MPLS were slightly higher. An acute congestion scenario is equivalent to our failure scenarios. What is interesting with the use of inter-domain LSPs is the predictability it brings in the QoS experienced by the traffic. It is a method for load balancing in the Internet. Moreover, knowing in advance the path (LSP) taken by the traffic will permit us to estimate bounds on the QoS parameters of interest.

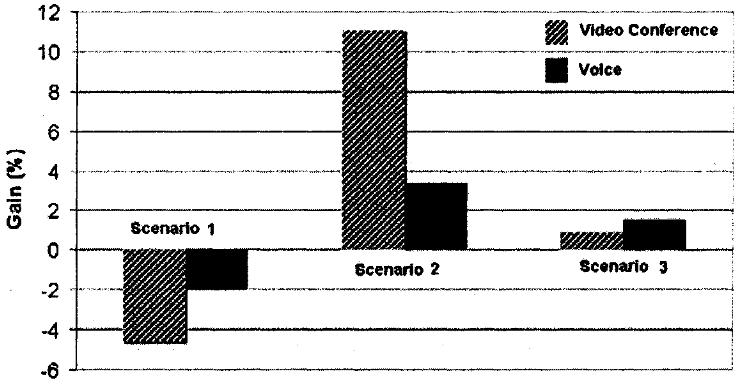


Fig.5: Gain in mean delay with inter-domain MPLS

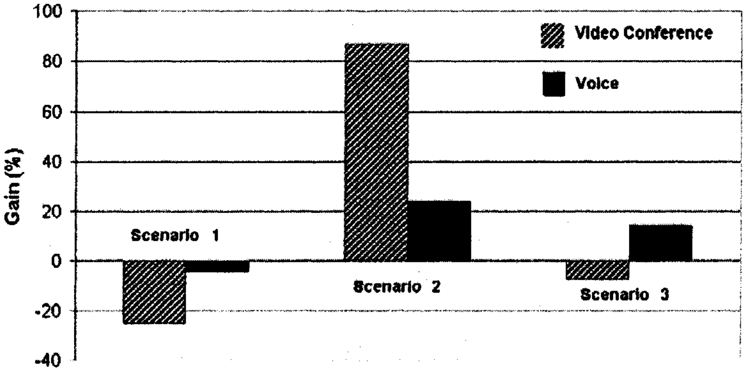


Fig.6: Gain in mean jitter with inter-domain MPLS

5 Conclusions

This paper consisted in refining and applying the already proposed extensions to RSVP-TE, to deploy MPLS across AS boundaries in order to achieve end-to-end control over the traffic in the Internet. Since Internet traffic crosses a few AS boundaries before reaching its destination, providing end-to-end QoS necessitates achieving end-to-end control of the traffic. MPLS is one of the best solutions for end-to-end control of the traffic and for end-to-end QoS provisioning, since it already serves these purposes inside ASs. Our future objectives are to define the signaling of the required QoS and to propose an end-to-end QoS provisioning architecture in the Internet.

References

1. D. Awduche, L. Berger, T. Li, V. Srinivasan, G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, Dec. 2001.
2. O. Bonaventure, S. De Cnodder, B. Quoitin, R. White, "Controlling the redistribution of BGP routes", April 2003. Work in progress, draft-ietf-grow-bgp-redistribution-00.txt.
3. O. Bonaventure, B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, "Interdomain Traffic Engineering with BGP", *IEEE Communications magazine*, Vol. 41, No. 5, May 2003, pp.122-128.
4. O. Bonaventure, B. Quoitin, "Common utilization of the BGP community attribute", June 2003. Work in progress, draft-bonaventure-quoitin-bgp-communities-00.txt.
5. R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, "Resource ReSerVation Protocol (RSVP)", RFC 2205, Sept. 1997.
6. M. Girish, B. Zhou, J.Q. Hu, "Formulation of the Traffic Engineering Problems in MPLS based IP Networks," *Proceedings of the Fifth IEEE Symposium on Computers and Communications (ISCC 2000)*, Antibes, France, July 4-6, 2000, pp. 214-219.
7. T. Okumus, J. Hwang, H.A. Mantar, S.J. Chapin, "Inter-Domain LSP Setup Using Bandwidth Management Points", *Proc. of IEEE Global Communications Conference Globecomm 2001*, Nov. 2001, San Antonio, TX, USA, pp.7-11.
8. P. Pan, "Scalable Resource Reservation Signaling in the Internet", PhD thesis, Columbia University, 2002, 164 pages.
9. C. Pelsser, O. Bonaventure, "Extending RSVP-TE to support Inter-AS LSPs", *2003 Workshop on High Performance Switching and Routing (HPSR 2003)*, June 24-27th, 2003, pp.79-84.
10. E. Rosen, A. Viswanathan, R. Callon, "Multi-protocol Label Switching Architecture", RFC 3031, Jan. 2001.
11. S.R. Sangli, D. Tappan, Y. Rekhter, "BGP Extended Communities Attribute", Aug. 2003. Work in progress, draft-ietf-idr-bgp-ext-communities-06.txt.
12. R. Zhang, J.P. Vasseur, "MPLS Inter-AS traffic engineering requirements", May 2003. Work in progress, draft-zhang-mpls-interas-te-req-03.txt.

AUTONOMIC SERVICE CONFIGURATION BY A COMBINED STATE MACHINE AND REASONING ENGINE BASED ACTOR

Paramai Supadulchai and Finn Arve Aagesen
NTNU, Department of Telematics, N-7491 Trondheim, Norway

Abstract: Service systems constituted by service components are considered. Service components are executed as software components in nodes, which are physical processing units such as servers, routers, switches and user terminals. A capability is an inherent property of a node or a user, which defines the ability to do something. Status is a measure for the situation in a system. A service system has defined requirements to capabilities and status. Because of continuous changes in capabilities and status, dynamic service configuration with respect to capabilities and status is needed. Software components are generic components, denoted as actors. An actor is able to download, execute and move functionality, denoted as a role. Configuration is based on the matching between required capability and status of a role and the present executing capabilities and status of nodes. We propose an approach for role specification and execution based on a combination an Extended Finite State Machine and a rule based reasoning engine. Actor execution support consisting of a state machine interpreter and a reasoning engine has been implemented, and has also been applied for a service configuration example.

1. INTRODUCTION

Service systems constituted by *service components* are considered. Service components are executed as software components in *nodes*, which are physical processing units such as servers, routers, switches and user terminals such as phones, laptops, PCs, and PDAs. Traditionally, the nodes as well as the service components have a predefined functionality. However, changes are taking place. Nodes are getting more generic and can have any kind of capabilities such as MP3, camera and storage. The software components have been also changed from being static components to become more dynamic and be able to download and execute different functionality depending on the need. Such generic programs are from now on denoted as *actors*. The name actor is chosen because of the analogy with the actor in the theatre, which is able to play different *roles* play defined in different *plays*.

To utilize the flexibility potential, the attributes of services, service components, software components and nodes must be appropriately formalized, stored and made available. As a first further step towards this formalization, the concepts *status and capability* are introduced.

Status is a measure for the situation in a system with respect to the number of active entities, the traffic situation and the Quality of Service. A *capability* is an inherent property of a node or a user, which defines the ability to do something. A capability in a node is a feature available to implement services. A capability of a user is a property that makes the user capable of using services. An actor executes a program, which may need capabilities in the node. Capabilities can be classified into:

- *Resources:* physical hardware components with finite capacity,
- *Functions:* pure software or combined software/hardware component performing particular tasks,
- *Data:* just data, the interpretation, validity and life span of which depend on the context of the usage.

The functionality to be played by an actor participating in the constitution of a service is denoted as its role. We use the role-figure as a generic concept for the actor which is playing a role. So services and service components are realized by role-figures. *Service configuration* is here the configuration of services with respect to the required capability and status of the roles.

The Role of an actor is defined in a manuscript, which consists of an EFSM *(Extended Finite State Machine)* extended with rule-based policies. Using a local *rule-based reasoning* engine adds the ability to cope with various situations more flexible than is possible by the pure EFSM. Actors can locally take place in the configuration and reconfiguration of the services, in which they are a part of. The reasoning engine is based an XET (XML Equivalent Transformation) rule-based language.

The work presented in this paper has been related to the Telematics architecture for Play-based Adaptable System (TAPAS) [2]. Section 2 discusses related work. Section 3 presents the model used for the combined EFSM and reasoning engine based actor. Section 4 gives a short presentation of the TAPAS architecture with focus on the elements relevant for the autonomic service configuration. Section 5 presents the data model. Section 6 presents a simple scenario where an actor actively participates in service reconfiguration. Section 7 gives a summary and presents our conclusions.

2. RELATED WORK

The mobility of service components have been dealt within a number of approaches. An example is the Intelligent Agent, which is the most related to our work. DOSE [4] is an agent-based autonomic platform that uses Semantic Web to come up with response to failures. However, the behavior of each service component must be fixed along with the moving codes that cannot be downloaded or changed. A technique to overcome this shortcoming has been proposed using Java Reflection [5]. The behavior of server components can be downloaded or changed based up on a reasoning mechanism. However, the reasoning mechanism itself cannot be downloaded or altered. In our approach, the behavior of service components and the rules used in the reasoning mechanism are downloadable and can be changed upon needs.

3. THE ACTOR MODEL

The actor role is defined as an Extended Finite State Machine (EFSM) extended with policies. The mechanism interpreting the manuscript is an EFSM interpreter extended with a reasoning mechanism. The data structure applied for the representation of an EFSM is shown in Figure 1. An EFSM contains the EFSM name, initial state, data and variables and a set of states. The state structure defines the name of the state and a set of transition rules for this state. Each transition rule specifies that for each input, the actor will perform a number of actions, and/or send a number of outputs, and then go to the next state. Actions are functions and tasks performed during a specific state: computation on local data, role session initialization, message passing, etc. The structure of the <ACTIONS> list specifies the name, the parameters and the classification of an action.

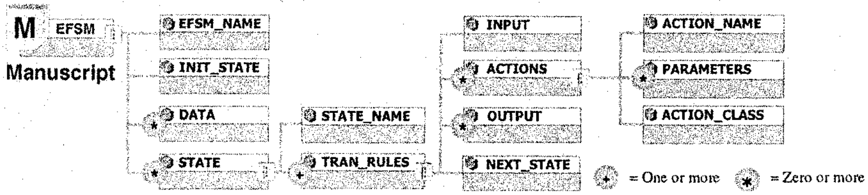


Figure 1. Data structure of EFSM-based manuscript

Rule-based reasoning is considered as a special type of EFSM action that executes policies. Policies are expresses in the XML Equivalent Transformation language (XET) [3]. The reasoning engine can directly operate and reason about XET descriptions.

The XET language is an XML-based knowledge representation, which extends ordinary, well-formed XML elements by incorporation of variables for an enhancement of expressive power and representation of implicit information into so-called XML expressions. Ordinary XML elements, XML expression without variables, are denoted as ground XML expressions. Every component of an XML expression can contain variables as shown in Table 1. Every variable is prefixed with '*T*var_' where *T* denotes its type.

Table 1. Types of XML variables

Type	Instantiation and examples
N	XML element or attribute names Ex: <Nvar_X>...</Nvar_X> can be instantiated to <div>...</div> or <span>...</span>
S	XML string Ex: <a name='Svar_Y'/> can be instantiated into <a name='http://...'/> or <a name='ftp://...'/>
P	Sequence of zero or more attribute-value pairs Ex: <p Pvar_Z='NULL'/> can be instantiated into <p/> or <p style='...'/>
E	Sequence of zero or more XML expressions Ex: <p>Evar_P</p> can be instantiated into <p/> or <p><div>...</div><br/><br/></p>
I	Part of XML expressions Ex: <Ivar_X><hr/></IvarX> can be instantiated into <body><hr/></body> or <hr/>

A rule is an XML clause of the form:

$$H, \{C_1, \ldots C_m\} \rightarrow B_1, \ldots B_n$$

where m, $n \geq 0$, H and B_i are XML expressions. And each of the C_i is a predefined XML condition used to limit the rule for a certain circumstances. This allows constraints modeling for a rule. Axioms are defined from one or more rule(s). The XML expression H is called the head of the clause. The B_i is a body atom of the clause. When the list of body atom is empty, such a clause is referred to an XML unit clause, and the symbol '→' will be omitted. Hence ordinary XML elements or documents can be mapped directly onto a ground XML unit clause.

The reasoning process begins with an XML expression-based query. An XML clause will be formulated from the query in form:

$$Q \rightarrow Q$$

XML expression Q represents the constructer of the expected answer which can be derived if all the body atoms of the clause hold. However, if one or more XML expression body atoms still contain XML variables. These variables must be matched and resolved from other rules.

A body from the query clause will be matched with the head of each rule. At the beginning, there is only one body Q. Consider a rule R_1 in the form:

$$R_1: H, \{C_1\} \rightarrow B_1, B_2$$

If the XML structure of the body Q of the clause and the head H of the rule R_1 match without violating condition C_1, the body Q will be transformed into B_1 and B_2. All XML variables in the head Q and the new bodies B_1 and B_2 of the query clause will be instantiated. The query clause will be in the form:

$$Q^* \rightarrow B_1^*, B_2^*$$

Where X^* means the one or more variables in the XML expression X has been instantiated and removed.

The transformation process ends when either 1) the query clause has been transformed into a unit clause or 2) there is no rule that can transform the current bodies B_i of the query clause. If the constructor Q is transformed successfully into Q_f that contain no XML variable, the reasoning process ends and a desired answer is obtained.

4. TAPAS ARCHITECTURE

"Adaptable service systems" are service systems that adapts dynamic to changes in both time and position related to Users, Nodes, Capabilities, Status and Changed Service Requirements. Adaptability can be modeled as a property consisting of 3 property classes: 1) rearrangement flexibility, 2) failure robustness and survivability, and 3) QoS awareness and resource control. The Telematics Architecture for Play-based Adaptable System (TAPAS) intends to meet these properties [2]. In analogy with the TINA architecture [6], the TAPAS architecture is separated into a system management architecture and a computing architecture as follows:

- The system management architecture is an architecture showing the structure of services and services components.
- The computing architecture is a generic architecture for the modeling of any service software components.

These architectures are not independent and can be seen as architectures at different abstraction layers. The system management architecture, however, has focus on the functionality independent of implementation, and the computing architecture has focus on the modeling of functionality with respect to implementation, but independent of the nature of the functionality.

4.1 Computing architecture

TAPAS computing architecture has three layers: the service view, the play view and the network view as illustrated in Figure 2. For details see [1].

A service system consists of service components and the network system consists of nodes. *The play view* is the intended basis for designing functionality that can meet the adaptability properties as defined above. The play view is founded on the theater metaphor introduced in Sec.1. TAPAS actors are software components in nodes that can download manuscripts. An actor that does not have a role assigned is denoted as a *free actor.* An actor playing a role in a manuscript is denoted as a *role figure.* A service system is constituted by a play, and leaf service component are constituted by role figures. A *role session* is the dialog between two executing role figures. A role figure can move between nodes and its role sessions can be re-instantiated automatically. This mechanism, however, is not the focus of this paper. It is referred to [7].

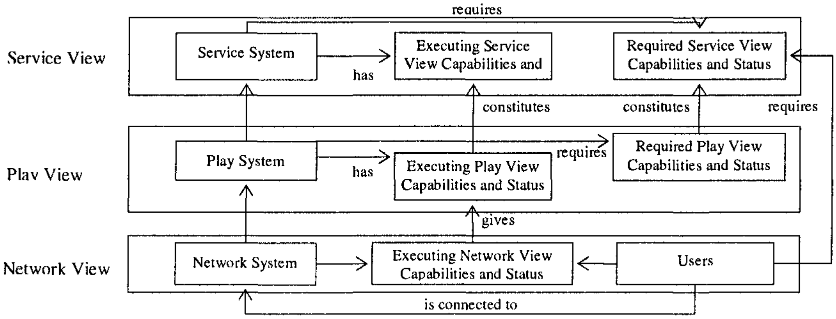

requires
Service View
Service System
has
Executing Service View Capabilities and
Required Service View Capabilities and Status
constitutes
constitutes
requires
Plav View
Play System
has
Executing Play View Capabilities and Status
requires
Required Play View Capabilities and Status
gives
Network View
Network System
Executing Network View Capabilities and Status
Users
is connected to

Figure 2. The TAPAS computing architecture

4.2 System management architecture

The main functionality components of the system management architecture are illustrated in the Figure 3. The primary service providing functionality comprises the ordinary services offered to human users.

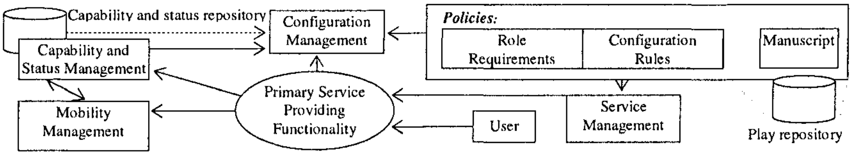


Figure 3. The TAPAS system management architecture

In addition, the architecture has two repositories: the Play repository and the capability and status repository and fours management components: Configuration, Service, Capability and status, and Mobility management.

The play repository stores *manuscripts* and *policies*, which are the required status and capability of a role as well as local configuration rules. Local configuration rules describe configuration and constraints of a role which must always be maintained. In addition, these rules define policies for handling of reconfiguration related events such as the decision of an actor to move a role when a failure happens. The capability and status repository stores *executing capability* and *status information.*

Configuration management makes the initial configuration and re-configures the service systems when needed. The Service management is responsible for deployment and invocation of services. Capability and status management registers, de-registers, updates and provide access to capability and status repository and the Mobility management handles the various mobility types.

To fulfill the failure robustness and survivability requirements, the architecture must be dependable and distributed. The proposed actor model creates a distributed configuration management by adding reasoning functionality to actors.

5. DATA MODEL

This section presents XML based approaches to the representation of the elements of the Play repository as well as the Capability and status repository.

5.1 Manuscript

A manuscript consists of EFSM-based behavior of individual roles. An XML-based EFSM given to an actor is executed by a state machine interpreter. A sample fragment of the XML-base manuscript is shown in Figure. 4.

```
<state name='ConnectionTimeout'>...</state>                      %
<state name='ConnectionLost'>                                    %
 <Transition name='RoleFigureMove'>                              %
  <input msg='RoleFigureMoveReq' source='*'/>                    %
  <action class='Reasoning' name='SearchFreeActor'>              %
    <param name='role_name' value='role1'/>                      %
  </action>                                                      %
  <output><variable name='Dest_Variable'/></output>              %
  <action class='Communication' name='PluginActor'>              %
   <param name='actorList' value='Dest_Variable'/>               %
   <param name='role_name' value='role1'/>                       %
  </action>                                                      %
  <next_state name='PlugoutPending'/>                            %
 </Transition>                                                   %
 ...                                                             %
</state>
```

After the state *ConnectionTimeout* is visited infinitely often, the actor playing this manuscript will move to *ConnectionLost* state. If there is an incoming message *RoleFigureMoveReq*, the actor will execute the *RoleFigureMove* transition and perform two subsequent actions. The first action uses the built-in reasoning machine to find out a free actor where the role should be moved to. The second action installs the role to a free actor suggested by the first action. At the end of the transition, the actor moves to *PlugoutPending* state and wait for a plugout message from the newly instantiated role figure.

Figure 4. Fragment of an example XML manuscript showing a transition of state *ConnectionLost*

SMI interprets the downloaded manuscript. SMI uses *action libraries.* Policy related actions are platform independent constraints expressed in XET (see Section 2). For non-policy actions, the actions are platform-specific (such as C++) or platform-independent (such as Java) executable codes from the local action library cache to execute the actions in the transition. If the required action libraries cannot be found, SMI will download the actions from an action library database.

5.2 Executing Capability and Status

Nodes possess particular Network View Capabilities and Status, from now on abbreviated as NV-capabilities and -status. They are represented in a network information model such as Common Information Model (CIM) or Universal Plug-and-Play. We have chosen the XML representation of CIM (CIM-XML) to implement our test systems.

Actors have Play View capabilities and status abbreviated as PV-capabilities and -status. The idea is to hide the complexity of the network view. PV-capabilities and -status of an actor are derived from one or more NV-capabilities and -status. PV-capabilities and -status are represented in Resource Definition Framework (RDF) [9], which can be used to either define pointers to NV-capabilities and -status or define derived PV-capabilities and -status from NV-capabilities and -status [8].

5.3 Policies

The policies comprises: role requirements, local configuration rules. These are modeled by the XET language (See Section 3).

5.3.1 Role requirements

Role requirements consist of PV-capabilities and -status required by a role. These PV-capabilities and -status are represented in RDF and XML variables.

5.3.2 Local configuration rules

The heads of the XET clauses identify components of the outcome of the configuration or reconfiguration, while the body describes the configuration, composition and dependency conditions. A sample local configuration rule is illustrated in Fig. 5.

```
<xet:Rule name='SearchFreeActor' priority='3'>
  <xet:Head>
   <tapas:Actor rdf:resource='Svar_ActorID'/>
 </xet:Head>
 <xet:Body>
  <xfn:FactQuery xfn:uri='ds://PV-Repository' xfn:mode='Set'>
   <tapas:Actor rdf:about='Svar_ActorID'>
    <tapas:connectivity rdf:resource='dbServer'>
     <tapas:connStatus rdf:resource='Status_Active'/>
     <tapas:connType rdf:resource='Svar_connType'/>
     Evar_otherConnProps
    </tapas:connectivity>
    <tapas:actorStatus rdf:resource='Status_FreeActor'/>
    Evar_otherActorProps
   </tapas:Actor>                              Query Expression
  </xfn:FactQuery>
  <xfn:StringIsMember xfn:string='Svar_connType'
    xfn:list='Secured SecuredWireless'>
 </xet:Body>
</xet:Rule>
```

Intuitively, this rule looks for free actors that have a secured connection with *dbServer*, which is a database server providing sensitive information. The head of the rule will be derived as answer(s) if both body atoms can be successfully executed.

Namespace *xfn* refers to built-in atoms providing mathematic operations and database query, etc. These atoms will not be further matched with other rules. *FactQuery* queries actors from the capability and status repository. The query expression simply ignores the order of XML elements when it is working in mode *"set"*. Some irrelevant PV-capabilities and -status of actors are ignored by using two E-variables. *Evar_otherConnProperties* and *Evar_otherActorProps*.

The actors must have *Status_FreeActor* as specified in the query expression. They must have only *active secured* or *secured wireless connectivity* with *dbServer1*, which will be checked by the builtin atom, *StringIsMember*.

Figure 5. An example XET clause to search for free actors

6. DEMONSTRATION

A scenario of a secured database system is considered as an example. A database server contains sensitive information and will automatically blocks incoming requests from nodes that could possibly have malicious software such as viruses or trojans.

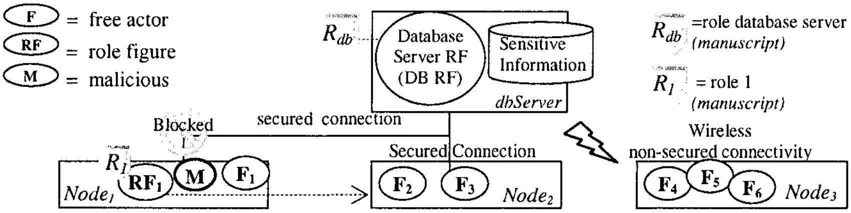


Figure 6. A sample scenario showing the survivability of a role figure.

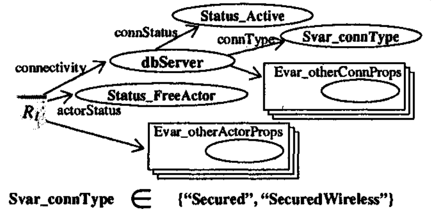
The goal of this demonstration is to show how a role figure in a blocked node can survive, move to other one other node, identified as harmless by the database server, and continue working with the database server. How the role figure proves itself as non-malicious software is not the focus and will not be further explained.

Fig. 6 illustrates a role figure RF_1 in Node 1, which is presently blocked by a database server role figure (*DB RF*). After RF_1 visits a state *ConnectionTimeout* infinitely often, it will move to *ConnectionLost* state. At *ConnectionLost*, *RoleFigureMove* transition will initiated by a

RoleFigureMoveReq message. The manuscript describing this transition has been presented in Figure 4.

6.1 R1 role requirement (required PV-capabilities and -status)

The PV-capabilities and -status required by the role R_1 are illustrated in Fig. 7. R_1 explicitly needs status *Status_FreeActor*. The connectivity between R_1 and *dbServer* must be a member of set {"Secured", "SecuredWireless"}. Actors trying to play R_1 may have other PV-capabilities and -status (as represented by *Evar_otherActorProps* and *Evar_otherConnProps*). These PV-capabilities and -status will be ignored by the reasoning engine.



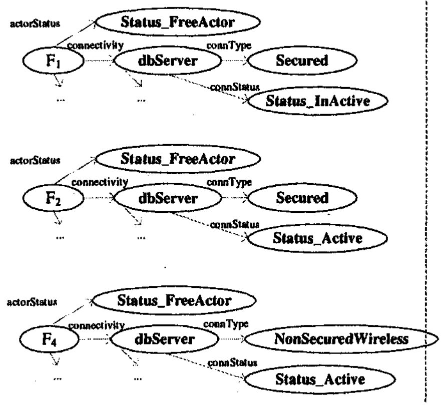
```
<tapas:Role rdf:about='R1'>
 <tapas:connectivity rdf:resource='dbServer'>
  <tapas:connStatus rdf:resource='Status_Active'/>
  <tapas:connType rdf:resource='Svar_connType'/>
  Evar_otherConnProps
 </tapas:connectivity>
 <tapas:actorStatus rdf:resource='Status_FreeActor'/>
 Evar_otherActorProps
</tapas:Actor>

<xfn:StringIsMember xfn:string='Svar_connType'
                    xfn:list='Secured SecuredWireless'/>
```

Figure 7. The required PV-capabilities and -status of the role R1

6.2 Offered PV-capabilities and -status

Fig. 8 shows the offered PV-capabilities and -status of actor F_1, F_2 and F_4. The PV-capabilities and -status of F_3 are identical to F_2 while the capabilities and status of F_5 and F_6 are identical to F_4. For lack of space, they will not be presented.

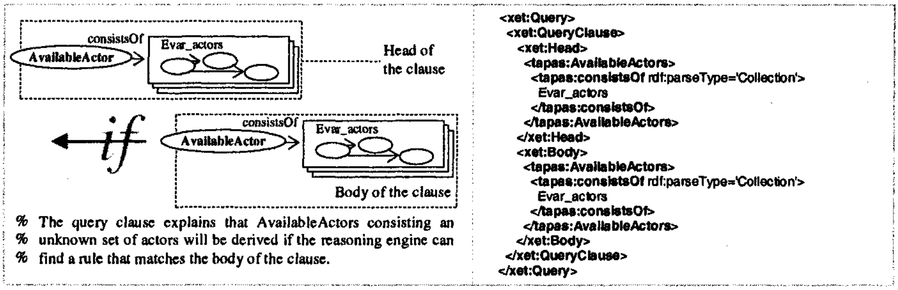


```
<tapas:Actor rdf:about='F1'>
 <tapas:connectivity rdf:resource='dbServer'>
  <tapas:connStatus rdf:resource='Status_Active'/>
  <tapas:connType rdf:resource='Secured'/>
  ...
 </tapas:connectivity>
 <tapas:actorStatus rdf:resource='Status_FreeActor'/>...
</tapas:Actor>
<tapas:Actor rdf:about='F2'>
 <tapas:connectivity rdf:resource='dbServer'>
  <tapas:connStatus rdf:resource='Status_Active'/>
  <tapas:connType rdf:resource='Secured'/>
  ...
 </tapas:connectivity>
 <tapas:actorStatus rdf:resource='Status_FreeActor'/>...
</tapas:Actor>
<tapas:Actor rdf:about='F4'>
 <tapas:connectivity rdf:resource='dbServer'>
  <tapas:connStatus rdf:resource='Status_Active'/>
  <tapas:connType rdf:resource='NonSecuredWireless'/>
  ...
 </tapas:connectivity>
 <tapas:actorStatus rdf:resource='Status_FreeActor'/>...
</tapas:Actor>
```

Figure 8. The offered PV-capabilities and -status of actor F1, F2 and F4

6.3 Query clause

The query clause in Fig. 9 is constructed from a query expression. As already explained in Section 3, the body of the clause will be initially matched with a configuration rule, which will be defined in the Section 5.4.



% The query clause explains that AvailableActors consisting an
% unknown set of actors will be derived if the reasoning engine can
% find a rule that matches the body of the clause.

```
<xet:Query>
 <xet:QueryClause>
   <xet:Head>
    <tapas:AvailableActors>
     <tapas:consistsOf rdf:parseType='Collection'>
      Evar_actors
     </tapas:consistsOf>
    </tapas:AvailableActors>
   </xet:Head>
   <xet:Body>
    <tapas:AvailableActors>
     <tapas:consistsOf rdf:parseType='Collection'>
      Evar_actors
     </tapas:consistsOf>
    </tapas:AvailableActors>
   </xet:Body>
 </xet:QueryClause>
</xet:Query>
```

Figure 9. Graphical notation of the query to search for available actors

6.4 Local configuration rules

Local configuration rules as illustrated in Fig. 10 indicate that the connection status and the connection type of the link between R_1 and *DB RF* must be maintained in a secured manner. The only QoS parameter defined here is *Svar_connType.*

6.5 The configuration result

The configuration result is shown in Fig 11. Based on the offered PV-capability and -status provided in Fig. 8 and the role requirement defined in Fig. 7, actors F_2 and F_3 are the most appropriate actors to play R_1.

The reasoning process is conducted by Native XML Equivalent Transformation reasoning engine (NxET) implemented as a Java-based action for the state machine interpreter (SMI). NxET is used by SMI to execute *SearchFreeActor* action defined in Fig. 5. The parameter *actorList* of the *PluginActor* action will be substituted with the available actors in Fig. 11. *PluginActor* will try to move R_1 to F_2 first. If the moving is not successful, *PluginActor* will try again with F_3. Subsequently, RF_1 will move to *PlugoutPending* state after R_1 has been successfully moved to either R_2 or R_3. At this state, R_1 will be plugged out from RF_1, which will become a new free actor.

```
<xet:Rule name='SearchFreeActor' priority='3'>
  <xet:Head>
   <tapas:AvailableActors>
    <tapas:consistsOf rdf:parseType='Collection'>
     Evar_actors
    </tapas:consistsOf>
  </tapas:AvailableActors>
  </xet:Head>
  <xet:Body>
   <xfn:SetOf xfn:mode='Set'>
    <xfn:Set>Evar_actors</xfn:Set>
    <xfn:Constructor>
     <tapas:Actor rdf:resource='Svar_ActorID'/>
    </xfn:Constructor>
    <xfn:Condition>
     <tapas:Actor rdf:resource='Svar_ActorID'/>
    </xfn:Condition>
   </xfn:SetOf>
  </xet:Body>
</xet:Rule>
<xet:Rule name='R1Requirement' priority='4'>
  <xet:Head>
   <tapas:Actor rdf:resource='Svar_ActorID'/>
  </xet:Head>
  <xet:Body>
   <xfn:FactQuery xfn:uri='ds://Play-Repository' xfn:mode='Set'>
    <tapas:Role rdf:about='Svar_RoleID'>
     Evar_properties
    </tapas:Role>
   </xfn:FactQuery>
   <xfn:FactQuery xfn:uri='ds://PV-Repository' xfn:mode='Set'>
    <tapas:Actor rdf:about='Svar_ActorID'>
     Evar_properties
    </tapas:Actor>
   </xfn:FactQuery>
   <xfn:MatchD xfn:mode='Set'>
    <Expression>
     <tapas:connectivity rdf:resource='Svar_resource'>
      <tapas:connType rdf:resource='Svar_connType'/>
      Evar_otherConnProps
     </tapas:connectivity>
     Evar_otherActorProps
    </Expression>
    <Expression>Evar_properties</Expression>
  </xfn:MatchD>
  <xfn:StringIsMember xfn:string='Svar_connType'
     xfn:list='Secured SecuredWireless'/>
 </xet:Body>
</xet:Rule>
```

```
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
```

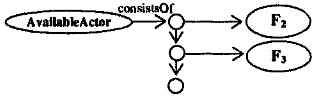
Rule *SearchFreeActor* will be matched with the body of the query clause defined in Section 5.3. After the matching, the body of the query clause will be re-written with *xfn:SetOf*, which is the only body atom of the rule. *xfn:SetOf* will to try to construct the list of available actors and add them into *Evar_actors* variable. Each members of *Evar_actors* will have the structure similar to the expression in *xfn:Constructor*. To actually instantiate the possible values for the constructor, the condition expression in *xfn:Condition* will be matched with other rules (clearly *R1Requirement*).

R1Requirement queries the R1 role requirement (required PV-capabilities and -status), which have been defined in Section 5.1. The rule again queries actors offering the same PV-capabilities and -status, which R_1 requires. The matching between required and offers PV-capabilities and -status are accomplished though the instantiation of variable *Evar_properties*. The actors queried from the capability and status repository needs *Status_Active* and *Status_FreeActor*. The actors can also have other PV-capabilities and -status because they are allowed by the role requirement.

The structure of PV-capabilities and -status of each actor will be matched with *xfn:MatchD* function so that PV-capability *connType* with a value *Svar_connType* can be inspected. Function *StringIsMember* verifies the instantiated value of *Svar_connType* to make sure that it is a member of the list "*Secured SecuredWireless*". Actors that do not offer secured or secured wireless connection will be filtered out. Only qualified one will be selected. *R1Requirement* can return many answers.

The answers returned by *R1Requirement* will be aggregated and added to *Evar_actors* list in the rule *SearchFreeActor*. The value of *Evar_actors* will be instantiated to the head of the query clause, which will be the answer of the reasoning process.

Figure 10. Local configuration rules in XET



```
<tapas:AvailableActors>
 <tapas:consistsOf rdf:parseType='Collection'>
   <tapas:Actor rdf:resource='F2'/>
   <tapas:Actor rdf:resource='F3'/>
 </tapas:consistsOf>
</tapas:AvailableActors>
```

Figure 11. RDF-based graphical notation and XML-serialization of the configuration result

7. CONCLUSION

This paper presents an approach to model the behavior of service systems by actors playing roles defined in manuscripts. The actor is a combination of an Extended Finite State Machine (EFSM) and a rule based reasoning engine.

A service system has defined requirements to capabilities and status. Because of continuous changes in capabilities and status, dynamic service configuration with respect to capabilities and status is needed. Configuration is based on the matching between required capability and status of a role and the present executing capabilities and status. Roles are allowed to be moved to increase failure robustness and survivability of a service system. This role mobility can be achieved through EFSM behavior. However, using a rule-based reasoning mechanism allows actors to use local configuration rules to take decisions based on the current executing capabilities and status. The actor model improves actor functionality, increases survivability and makes the configuration management distributed.

Generic actor execution support consisting of a state machine interpreter and a reasoning engine has been implemented and applied for the presented example. All capability and status related data as well as actor behavior is based on XML representations, with exceptions of the EFSM actions. Normal EFSM actions are platform-specific (such as C++) or platform-independent (such as Java) executable codes while reasoning-based EFSM actions are XML-based. The reasoning engine is based on Native XML Equivalent Transformation.

REFERENCES

[1] Aagesen, F.A., et al., Configuration Management for an Adaptable Service System, *IFIP Open Conference on Metropolitan Area Networks Architecture, protocols, control, and management*, Viet Nam, 4/2005

[2] Aagesen, F.A., et al., On Adaptable Networking. *ICT'2003, Assumption University*, Thailand, 4/2003.

[3] Anutariya, C., et al., An Equivalent-Transformation-Based XML Rule Language. *Int'l Workshop Rule Markup Languages for Business Rules in the Semantic Web*, Italy, 6/2002.

[4] Bonino, D., et al., An agent based autonomic semantic platform, *Proc. Int'l Conf. on Autonomic Computing 2004*, 5/2004.

[5] Huang, G., et al., Towards autonomic computing middleware via reflection, *Proc. of the 28th Annual International COMPSAC 2004*. 9/2004.

[6] Inoue, Y., et al., The TINA Book. A Co-operative Solution for a Competitive World. Prentice Hall, 1999.

[7] Shiaa, M.M., Mobility Support Framework in Adaptable Service Architecture. *IEEE/IFIP Net-Con'2003*, Oman, 10/2003.

[8] Supadulchai, P., Aagesen, F.A., An Approach to Capability and Status Modeling, *NIK 2004*, Norway, 11/2004.

[9] World Wide Web Consortium, Resource Description Framework (RDF): Concepts and Abstract Syntax, Available online at http://www.w3.org/TR/rdf-concepts/.

SELF-MANAGEMENT IN AMBIENT NETWORKS FOR SERVICE COMPOSITION

Lawrence Cheng[1], Roel Ocampo[1], Alex Galis[1], Robert Szabo[2], Csaba Simon[2], Peter Kersch[2]

[1] *University College London, Electrical Engineering Department, Torrington Place, London, WC1E 7JE, UK {l.cheng, r.ocampo, a.galis}@ee.ucl.ac.uk:*

[2] *Budapest University of Technology and Economics, Department of Telecommunication and Media Informatics, Magyar Tudosok krt. 2., 1117, Budapest, Hungary {szabo, simon, kersch}@tmit.bme.hu*

Abstract: This paper describes the concepts and challenges of self-managing management-layer network composition and service composition in Ambient Networks. A set of requirements are identified. This paper describes the concept of Ambient Virtual Pipe (AVP), which is an autonomic, secure, QoS-assured, self-adapted context aware management service overlay network that provides a secure and QoS-assured environment for AN service composition. The AVP is supported through a programmable platform, and is capable of dynamic deployment of new management services.

Key words: Ambient networks; context-awareness; programmable techniques; self-management; service composition.

1. INTRODUCTION

The EU-IST Ambient Networks (AN) project [4] focuses on the development of novel networking concepts and systems that support a wide range of user and business communication scenarios beyond today's fixed, 3rd generation mobile and IP standards. The concept of Ambient Control Space (ACS) [5] is the centre of the project. The ACS is responsible for the management of the underlying data transmission capabilities. A complex set of interdependent control functions form the ACS. The management of AN is conducted by the Domain Manager Control Function. This management function works consistently and autonomically with other control functions being developed in the AN project. Details of ACS and Domain Manager Control Function can be found in [4][5].

AN management systems support the *composition* and *cooperation* of heterogeneous networks, on demand and transparently. Composition and cooperation must be achieved without the need of manual (pre or re)-configuration or off-line negotiations between network operators. Thus, AN management systems must be dynamic, distributed, self-managing and responsive to the network and its ambience [6]. The composition of heterogeneous ANs means an Ambient Network is able to dynamically compose with several other Ambient Networks. Co-operations between Ambient Networks could potentially belong to separate administrative or economic entities. Hence, Ambient Network composition provides network services across a set of ANs which were originally independent of each other in a cooperative way. The Ambient Network Interface (ANI) provides the facility of co-operation across different Ambient Networks. It is through the ANI that different management systems and network elements of ANs may communicate and co-operate with each other. A composed and cooperating AN(s) enable a user to access transparently the services offered by other Ambient Networks (that are previously independent of each other) via the Ambient Service Interfaces (ASI). Figure 1 shows the concept of composition and co-operation, and the logical location of ANI and ASI in AN(s) [5].

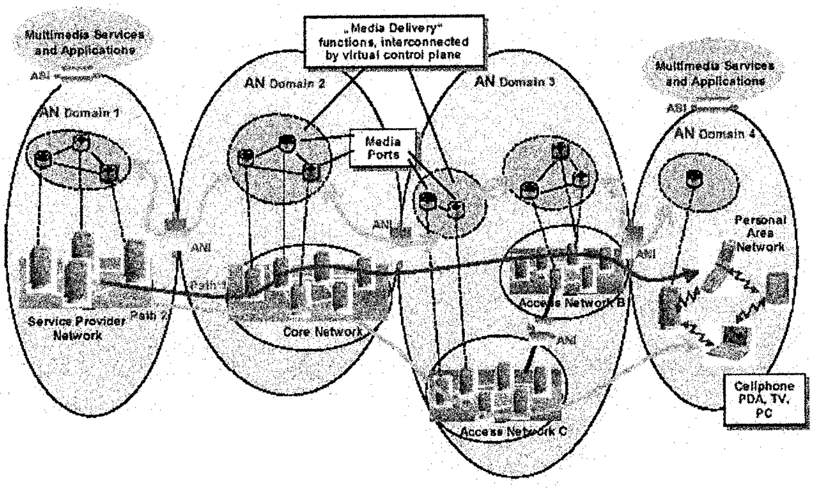


Figure 1. ASI and ANI in AN(s)

This paper starts with the discussion on the requirements of AN network composition in management point of view; followed by a description on a solution towards self-managing management-layer network composition and service composition in ANs. The proposed solution is known as the *Ambient Virtual Pipe (AVP),* which creates a management service overlay network for dynamic deployment of new management service over composed AN.

2. AN COMPOSITION REQUIREMENTS

Network composition is one of the major concepts of Ambient Networks. An AN may consist of many individual smaller ANs. The smaller ANs were composed through network composition. Through network composition, the sharing of network resource such as inter-network connectivity or network storage are negotiated during network composition according to policies. Note that because ANs are mobile, they may compose and decompose dynamically. As a result of network composition, end users are capable of being connected, and connecting to any network instantly. This is known as network-layer composition in this paper. From management point of view, AN network composition refers to the instant negotiation and enforcement of a new Service Level Agreement (SLA) between network resources under composition for the provisioning of an IP service. From a business point of view, network composition can be viewed as a temporary agreement among independent networks. A common business goal is achieved by the collaboration of agreements. The diversity and complexity of the market are matched by the temporary agreement. Thus the capability of rapid reaction to the dynamically-changing demands of today's markets is improved. It is obvious that for scalability and performance issue, the process of network composition should be as transparent as possible. One of the major challenges of compositions is currently there is a lack of support for automatic creation and administration of composed/composing service networks. Automatic service composition refers to the discovery of adequate ANs and their services, negotiation among them, the definition of business relations, the collection of configuration information and requirements, and the reservation of appropriate resources in the network infrastructure. AN network-layer composition is discussed in details in [8].

The management challenge of AN network composition is that the management systems of individual networks must also be composed during network composition for the purpose of consistency. This is known as management-layer network (de)composition in this paper. The requirements for AN management-layer network composition is as follow. Detail discussions on AN management challenges can be found in [5]:

a) The composition mechanism should be *performance-wise low cost*, *robust* and *scalable*. Performance and scalability are concerned because potentially a large number of ANs may compose and decompose at a given time (so as their management systems). Robustness is needed because the underlying ANs may compose and decompose dynamically as the ANs join and leave arbitrarily (so as the management-layer).

b) The network management systems of the underlying ANs that are composing or decomposing may be heterogeneous. The composition mechanism should be *flexible* to overcome heterogeneity.

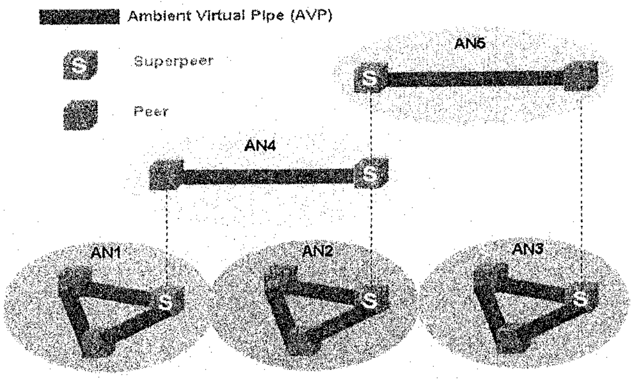
c) *Service-oriented* composition mechanism is needed in order to support new management services running in the composed management system.

d) Due to the potential large number of AN composition and decomposition (due to AN nodes joining or leaving the AN arbitrarily), the composition mechanism should be *autonomic*.

Conclusively, a self-managed management system is needed in AN to support autonomic management-layer composition and service composition for dynamic deployment of AN services across heterogeneous composed/composing ANs.

3. AMBIENT VIRTUAL PIPE (AVP)

It was discussed in early section that the AVP is an autonomic, secure, self-adapted, and context aware management service overlay network across composed AN nodes. This management service overlay network is self-adapted according to change in underlying network context information in order to support the dynamic deployment and execution of new AN (management) services in the composed management system, thus provides a dynamic, secure and QoS assured channel for P2P management traffic. A highly dynamic and flexible information infrastructure is needed in order to support new management service deployment over a composed AN management domain. This information infrastructure must be capable of providing secure and reliable connectivity across heterogeneous networks with guaranteed quality on demand. It may be arguable that conventional solutions like the currently available Virtual Private Networks (VPNs) may be used to provide QoS guarantees to networks. However, the major drawback of the today's VPNs is their low flexibility to quickly adapt to the changing requirements. A programmable [11], flexible, and network contextaware information infrastructure is therefore needed with guaranteed QoS in the composed AN management domain. This infrastructure is the AVP.



The AVPs are shown in Figure 2. The AVP provides a secure and QoS-assured channel for protecting P2P management information exchanged between distributed AN management entities in composed ANs, and the AVP creates an environment where new AN management services to be launched and executed i.e. service composition across heterogeneous ANs.

This secure, QoS-assured, management service network overlay is essential for both management-layer composition in AN as well as new AN management service deployment and execution. As discussed earlier, AN network composition is carried out through negotiation, AN management systems composition is also carried out through negotiation between heterogeneous management systems. Superpeer election is conducted through negotiations between peers. Distributed management information must be shared securely and in a QoS-assured fashion to enable negotiation to take place (hence management-layer and service composition). The AVP provides a suitable environment in which all these management negotiations may take place. Traditionally, a network domain administrator is capable of managing the administrative tasks within its own administrative domain. End users within a particular administrative domain may request for services that are served with some level of service guarantee from its own network domain administrator. However, inter-domain service management is complex, this is due to the heterogeneity of the administrative environment and the underlying network elements of different administrative domains. The idea of AN management-layer network composition through the P2P management system and the creation of a management service overlay network among the peers provides a mean to achieve inter-domain service management in a secure, QoS-assured, and self-adaptable environment. The network context aware capability of AVPs provide the necessary QoS and resource assurance and security for protecting the management traffic within the management service overlay network in a *dynamic* fashion. The creation of AVPs and its capabilities (such as security and contextawareness) are supported by a flexible and programmable infrastructure (see later section on implementation).

Note that AVPs are not restricted to provide a secure and QoS assured management service overlay network for management services. The capabilities of AVPs make it potentially ideal for the dynamic deployment of user specific services across heterogeneous ANs. For instance, with the existence of AVP, it is possible for a management entity in a particular AN to deploy its own services (that are not available in other ANs) in another (composed) AN. As discussed, the capabilities of AVPs are supported through a programmable platform. A set of requirements for AN programmability were defined in [5][7]: rapid development of new management services replaces slow manual configuration; customisation of existing management service features; scalability and cost reduction in AN network and service management; independence of AN network equipment manufacturers; information AN network context and service integration. The dynamic creation of AVP is achieved by dynamically injecting active code to desired peers in order to instantiate AVPs. The active code carries executable programs which result in security association implementation between peers. QoS in AVP is assured through the injection of active code to dynamically prioritising AVP traffic flow. DINA [2] is used as a programmable platform in AN to support AVP provisioning. Figure 3 shows the actual deployment of AVP through a programmable network. Details of the AVP implementation are discussed in the next section.

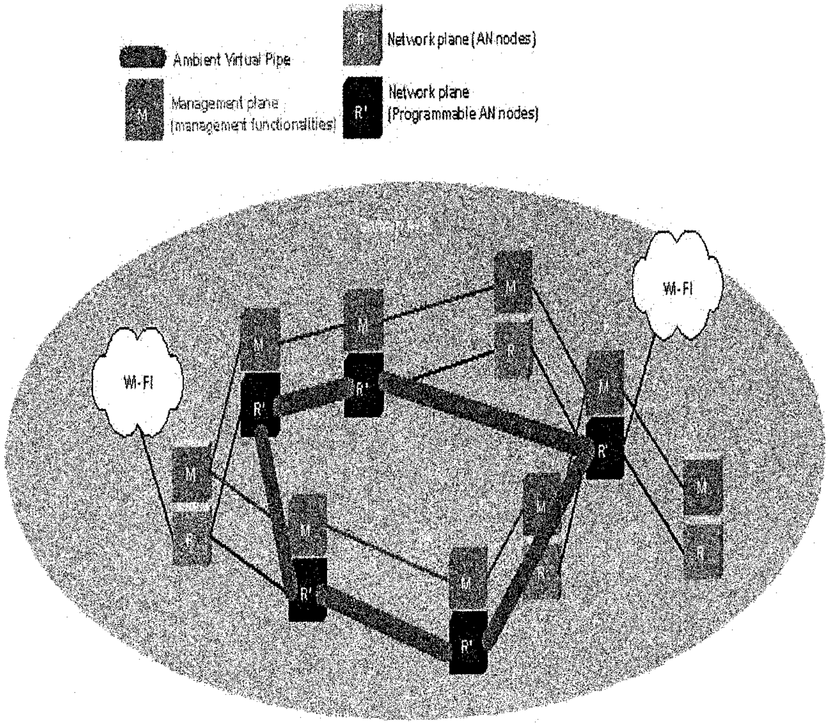


Figure 3. AVP Support through a Programmable Structure

4. IMPLEMENTATION OF AVP

The *AVP Managers* are responsible for managing AVPs .The AVP Managers are distributed across desired participating AN nodes in the composed AN network. The ContextWare components of AVP i.e. CMSs are responsible for monitoring real-time network context information, and are responsible for providing pointers to the distributed network context information that can be retrieved by the AVP Manager. Details of the ContextWare components of AVP are presented in [9]. Retrieving network context information by the AVP Managers subsequently triggers SNMP traps. The traps are then processed by sub-components of the AVP Manager according to pre-defined policies. The traps indicate the type of dynamic provisioning and self-adaptation to be deployed on the management service overlay network.

As discussed earlier, the autonomic establishment of AVP is done by using DINA [2][11] active packets. Active packets are also used for security provisioning and QoS provisioning of AVPs. The AVP QoS provisioning is an example of its capability of self-adaptation. An interface is provided between the AVP Manager and the Linux iptables and tc mechanisms for AVP internal flow marking and classification. This interface causes the AVP flow to be internally marked within the node using iptables. The flows are then classified by tc into specific classids. Bandwidth is allocated to each flow by setting the bandwidth of the associated queuing discipline (qdisc) of the tc classid. AVP security provisioning is needed in order to protect the authenticity, integrity and confidentiality of the management data transmitted in the AVPs, and also the active control packets that are transmitted and executed across AN nodes. The AVP Manager currently uses Freeswan IPSec and therefore supports IKE and IPSec. AVP supports hop-to-hop protection, which is needed for protecting the authenticity, integrity and confidentiality of active packets. The discussion and implementation of hop-to-hop protection for active packets is discussed in details in another paper [3] written by the author of this paper. Note that in the current implementation, as a proof of concept, a unicast IPSec structure is assumed. The security of multicast traffic is being investigated in [10]. The context awareness and capability of self-adaptation of AVP were demonstrated through various aspects. Firstly, when two (or more) ANs compose, AVP will be automatically established between the peers (of the two composing ANs). As soon as a composed AN is formed, a Superpeer is elected by the P2P management system (of the peers within the AN). Note that AN nodes are regarded as peers in AN. Peers are organised by a P2P management system in a hierarchical structure for scalable management [1]. A Superpeer is an elected peer of which is responsible for inter-AN communications. The election process of Superpeer is described in [1]. The establishment of AVP between Superpeers results in a new overlay network on top of the other peers. The selection of new Superpeer of this new overlay is done by negotiation between the two Superpeers' P2P management system, which is conducted securely and in a QoS assured fashion through the AVP. The activation of the secure channel is done by active packets. It should be note that the AVP Manager (through the use of IKE) is responsible for negotiating with peers on security associations (SAs) establishment. As soon as the CMSs detects a new peer has joint the AN, SAs are negotiated automatically. The CMSs also reports when an AN (Superpeers and/or peers) has left, the AVP will adapt itself when decomposing i.e. obsolete the established SA. This is an example of network contextawareness. Another example of self-adaptation is that once a new overlay is created, the AVP Managers retrieves network context information from the CMSs for QoS self-adaptation. For instance, when an AN composes with another AN, and the latter AN generates a large amount of management traffic, then the AVP

Manager will self-adapt the AVP bandwidth (in the original AN) through the use of active technologies as described in [9]. In this way, the QoS of the AVP in the original AN is assured.

5. CONCLUSION & FUTURE WORK

In this paper we presented the architecture and the key concept of Ambient Networks namely the management-layer network composition for service composition. A set of requirements were listed. We have identified the management challenges of network composition and service composition. A solution was then presented.

AN nodes are organised in peer groups and are placed in a hierarchical order, through a P2P management platform. The operations of P2P management system results in a hierarchically structured management overlay network aligned with the physical network structure. The hierarchy of management overlays is developed in accordance with network composition strategies. This can be viewed as a topological resource composition which structures the network resources and their topology according to dynamic composition rules/policies.

We then presented the concept of AVP which is an autonomic, secure, QoS-assured, self-adaptable, and contextaware management service overlay network that is needed to overcome the service composition challenges. This management service overlay network is created dynamically between AN management entities in order to provide a secure and QoS assured means of communication channels between management entities in composed ANs, as well as a secure and QoS-assured environment in which new AN (management) services may be deployed and executed. The instantiation of AVPs is supported through a flexible, scalable, and programmable infrastructure deployed among the peers. Through AVPs, it is now possible to transport management traffic and carry out negotiations between the management entities in the P2P management system in a secure and QoS assured way. The AVP also provides a secure and QoS-assured environment in which new services across heterogeneous ANs may be deployed. The AVP is self-adapted and contextaware, that it may automatically adjust its behaviour according to changing network context. Lastly, the implementation of AVP and its deployment were presented. The next stage of implementation is to refine the current implementation of the AVP Manager. Instead of unicast IPSec, a multicast alternative should be deployed should the management traffic is multicast. The SA establishment between peers must also be refined. This is because there may be a lack of a trusted third party in wireless domains. Performance results will be analyzed to prove the practicability of the presented solution.

ACKNOWLEDGEMENT

This paper describes work undertaken in the context of the Ambient Networks - Information Society Technologies project, which is partially funded by the Commission of the European Union.

REFERENCES

1 C. Simon, et al., Peer-to-peer management in Ambient Networks, poster in 14th IST Mobile & Wireless Communications Summit (2005).
2 D. Raz, et al., An Active Network Approach for Efficient Network Management, Lecture Notes in Computer Science 1653 Springer (1999), ISBN 3-540-66238-3.
3 L. Cheng, et al., Strong Authentication for Active Networks, IEEE-Softcom (2003).
4 Ambient Networks (2005), http://www.ambient-networks.org.
5 Brunner, M., et al., Ambient Networks Management Challenges and Approaches, ISBN 3-540-23423-3, Springer- Verlag Lecture Notes in Computer Science - IEEE MATA (2004).
6 Galis A., et al, Ambient Network Management – Technologies and Strategies, AN Deliverable 8.1 (2005), http://www.ambient-networks.org.
7 Jorge Andres, et al., R8-2 Report: Description of concept and scenarios for network composition management and self-management, (unpublished), Ambient Networks project internal report, 2004.
8 C. Kappler, et al., A Framework for Self-organising Network Composition, WAC (2004).
9 R. Ocampo, et al., ContextWare Support for Network and Service Composition and Self-Adaptation, to appear in IEEE-MATA (2005).
10 G. Selander, et al., Ambient Networks Intermediate Security Architecture, (2005), Deliverable 7.1, http://www.ambient-networks.org.
11 Galis A., et al., Programmable Networks for IP Service Deployment, Artech House Books, ISBN 1-58053-745-6; pp.450.

AMAPOLA: A SIMPLE INFRASTRUCTURE FOR UBIQUITOUS COMPUTING*

G. Navarro[1], J. Peñalver[1], J.A. Ortega-Ruiz[2], J. Ametller[1], J. Garcia[1], and J. Borrell[1]

[1] *Dept. of Information and Communications Engineering,*
Universitat Autonoma de Barcelona, 08193 Bellaterra, Spain
{gnavarro,jpenalver,jametller,jgarcia,jborrell}@ccd.uab.es

[2] *Institute for Space Studies of Catalonia,*
80034 Barcelona, Spain
jao@gnu.org

Abstract In this paper we present a simple framework for the management of entities in ubiquitous computing and ad-hoc networks. It provides mechanisms to identify entities, create and manage groups, and a simple management mechanism to allow the coordination of several entities. The framework is called AMAPOLA, and is built on top of a popular multiagent systems (JADE), although, its simplicity makes it suitable for any kind of environment. The framework provides an modular API, which is easy to use for programmers.

1. Introduction

The current development of computer systems is leading to a situation where the number of processors and computer networks is becoming more and more pervasive. Nowadays, there are processors embedded in lots of everyday devices. From personal computers, laptops, PDAs, and mobile phones, to refrigerators, heaters, coffee machines, or toasters. Furthermore, these devices can be interconnected through computer networks. The increased research on wireless and ad-hoc networks is making possible to have cheap networks at home, at the office or even at the streets.

One of the problems that pervasive computing introduces is the management of all those devices interacting one with another (Sloman, 2001), and the security implications of this management. A desired property of pervasive

*This work has been partially funded by the Spanish Ministry of Science and Technology (MCYT) though the project TIC2003-02041.

computing systems is self-management. Self-management is the ability for those systems to manage themselves with a minimum human intervention. For example, to tune and set up the configuration to make the system work optimally, to adapt the system to changing workloads, to detect potential attacks against the system itself, etc.

This is one of the reasons why multi-agent systems are becoming very popular in pervasive computing. A software agent is an autonomous entity that can interact and perceive the context of its own execution. Hence, it is a clear candidate to build self-managing systems in pervasive computing. A problem of multi-agent systems is that sometimes they present too much complexity for embedded devices. Most of the mechanisms used, for instance, to make up coalitions in agent systems, are quite complex and may not be suitable for some constrained environments.

In this paper we present a simple framework for the management of entities in pervasive computing and ad-hoc networks. It provides mechanisms to identify entities, create and manage groups, and a simple management mechanism to allow the coordination of several entities. The framework is called AMAPOLA (simple Agent-based MAnagement for Pervasive cOmputing). Although it is originally based on a multiagent system its simplicity makes it suitable for any kind of environment. The framework provides an modular API, which is easy to use for programmers.

In Section 2 we describe the motivations behind the AMAPOLA framework. Section 3 describes how identities and groups are managed in AMAPOLA, and Section 4 describes the simple management protocols. We give some high level details of the implementation of the framework in Section 5. Finally, Section 6 concludes the paper.

2. Related Work and Motivations

Despite the popularity of ubiquitous computing and the growing research initiatives, many of the current frameworks, systems, and prototypes lack scalability and are tied to third party proprietary solutions (Helal, 2005). On the other hand most proposals are also tied to specific hardware designs, making it difficult to reuse existing appliances and applications, and fail to provide a generic framework for ubiquitous computing spaces.

Some projects like Smart-Its (Holmquist et al., 2004) provide some generic programmable framework although it relies on an specific hardware architecture. An important contribution is the recent *Framework for Programmable Smart Spaces* project at the University of Florida. This project, presents a middleware architecture intended to be applicable to any pervasive computing spaces (Helal et al., 2005), which relies on the *Open Service Gateway initiative*

(OSGi) framework. There is no doubt that it is a good step, but we think some applications may need a simpler approach.

The use of agent technology in ubiquitous computing has been motivated by the autonomy and AI applications that multiagent systems can introduce in ubiquitous computing (3ap,).

AMAPOLA provides a novel approach for dealing with entities in ubiquitous computing environments, in a simple way. The main key points of AMAPOLA is simplicity, security, and easy of use. Security is built into the system beginning for how this entities are identified. As we will see the implementation of the framework makes it very easy to use for developers to program applications in ubiquitous environments, we also provide tools to help the programmers in their tasks.

All the information used by AMAPOLA is expressed using the *Secure Assertion Markup Language* (SAML) (S. Cantor, J. Kemp, R. Philpott and E. Maler, ed., 2005), which provides a popular standard XML-based framework for exchanging security information between online business partners. Security information is exchanged in form of *assertions*. Broadly speaking an assertion has an *issuer*, a *subject* or *subjects*, some *conditions* that express the validity specification of the assertion, and the *statement* (authentication, authorization decision, or attribute). The assertion may be signed by the issuer. SAML also provide query/response protocols to exchange assertions and bindings over SOAP and HTTP.

3. Identities and Group Management

The AMAPOLA framework uses a naming schema, which is influenced by the distributed local name system of the *Simple Public Key Infrastructure/Simple Distributed Security Infrastructure* SPKI/SDSI (Ellison et al., 1999). Each entity in AMAPOLA is known as *poppy*[1]. A *poppy* can be any piece of software taking active part in the framework, not only mobile and static agents, but also client applications directly controlled by a human.

Each poppy is uniquely identified by the *pID* (poppy ID). In reference to the pID, we can find two types of poppies:

***Strong* poppy** (or simply, poppy) The *pID* is a public key. Entities with the ability to perform asymmetric cryptographic operations, have a pair of cryptographic keys. The public key acts as the identifier of the poppy. In order to make it more manageable one can use the *hash* of the public key as an abbreviation for the public key. It is important to note that given the properties of cryptographic keys, it is commonly assumed the

[1] AMAPOLA means poppy in Spanish.

uniqueness of the public key, thus, we can assume that this kind of *pID* is globally unique.

***Weak* poppy** : The *pID* is the hash of an object. For entities not capable of carry out asymmetric cryptography operations, the identifier is computed as the *hash* of the entity code. If for some reason it is not possible to obtain the hash of the poppy's code, the hash of a nonce (a random byte array) is used. In both cases, and given the properties of the *hash* functions, we assume that the *pID* will be unique.

Each poppy has an associated local name space called *name container*. The name container has entries: `(<entity>,<local-name>)`, where *entity* corresponds to the poppy for whom the local name is being defined, and *local-name* is an arbitrary string. The *entity* may be specified as a *pID* or as a *fully qualified name* (see below).

For example, consider a poppy with a *pID* PK_0, which interacts with another one with *pID* PK_1 and wants to name it *partner*. The name container of the first poppy will have an entry of the form: `(`PK_1`, partner)`. Now on, the poppy PK_1 can be referenced by the name *partner* in the local name space of PK_0. An important issues is that a third parties can make a reference to a name defined in other name containers through a *fully qualified name*. A name container is identified by the *pID* of the owner, so the fully qualified name "PK_0 *partner*" makes reference to the name *partner* defined in the name container of PK_0 (which is PK_1). Intuitively one could say that PK_1 is PK_0's partner.

Name Assertions

Entries of a name container can be made public to rest of the world. This is specially relevant for groups and roles (see Section 3). In AMAPOLA, a local name may considered as an attribute associated to the corresponding poppy.

A name container entry can be expressed as a SAML assertion, where the issuer is the owner of the name container, the subject is the principal and the name is expressed as an *AttributeStatement*. We denote such an assertion as:

$$\{(PK_1\ ,\ \mathit{partner})\}_{PK_0^{-1}}$$

where PK_0^{-1} denotes the private key corresponding to the public key PK_0, which digitally sings the assertion determining the issuer or the owner of the name container where the name is defined. The assertion may also contain validity conditions, which are not shown for clarity reasons. As a consequence of the need for a digital signature, only strong poppies can issue name assertions. If a weak poppy needs to publish a local name from its name container, the assertion will have to be certified by another trusted strong poppy (for example one of the *holders* of the poppy, see Section 4).

Group Management

The AMAPOLA naming schema, makes it very easy for a poppy to create groups or roles. For instances, a poppy PK_{adm} can create a group *friends* with members PK_a, PK_b and PK_1 (recall the previous example), with the following name assertions:

$$\{(PK_1, \textit{friends})\}_{PK_{adm}^{-1}}$$
$$\{(PK_2, \textit{friends})\}_{PK_{adm}^{-1}}$$
$$\{(PK_0\ \textit{partner}, \textit{friends})\}_{PK_{adm}^{-1}}$$

This naming schema can also support role or group *hierarchies*, by means of group inclusion. This allows for the introduction of authorization schemas, and access control systems such a *Role-based Access Control* (RBAC). In order to do it one can declare a group as member of another group. For example, consider the role *family*, which is a super-role of *friends*. That is, members of *family* also have the attributes (permissions, authorizations, etc.) associated to *friends*. And at the same time members of the role *family* are also members of the role *friends*. This may be expressed as:

$$\{(PK_{adm}\ \textit{family}, \textit{friends})\}_{PK_{adm}^{-1}}$$

We differentiate between three types of group management based on the *leader* of the group. The *leader* is the owner of the name container where the group is to be defined. Depending on how this leader is set, there may be:

***Single-leader* group** : this is the common scenario where a single leader creates and manages a group. The way to do it is the one discussed in the previous example.

***Set-leader* group** : in this case there is a set of users entitled to manage the group. As an example, imagine that there are three poppies: PK_0, PK_1, and PK_2, and want to be the set of leaders for the group *intellcomm*. To do that, the poppies may generate the assertions listed in Table 1.

Table 1. *Set-leader* group management example.

PK$_0$	**PK$_1$**	**PK$_2$**
$\{(PK_0\ \alpha,\ \textit{intellcomm})\}_{PK_0}$	$\{(PK_1\ \alpha,\ \textit{intellcomm})\}_{PK_1}$	$\{(PK_2\ \alpha,\ \textit{intellcomm})\}_{PK_2}$
$\{(PK_1\ \alpha,\ \alpha)\}_{PK_0}$	$\{(PK_0\ \alpha,\ \alpha)\}_{PK_1}$	$\{(PK_0\ \alpha,\ \alpha)\}_{PK_2}$
$\{(PK_2\ \alpha,\ \alpha)\}_{PK_0}$	$\{(PK_2\ \alpha,\ \alpha)\}_{PK_1}$	$\{(PK_1\ \alpha,\ \alpha)\}_{PK_2}$

The members agree upon a random value α, enough large to insure no collision with names already listed in the name containers of each poppy. Each member issues a name assertion binding the group *intellcomm* to the random value, and then, they cross-certificate the α defined in each name container.

The use of α ensures no collision with names already defined. Each poppy can now manage the membership of the group, and even add new leaders either through simple inclusion or adding it to the initial set (this last operation requires the approval of the whole set of leaders since they have to cross-certificate the new one).

***Threshold-leader* group** : In this case, a group of n poppies agree upon creating a group, but in order to add new members to the group, a subset of k leaders has to agree ($k \leq n$). In order to do it, we use a (k, n)-*threshold scheme* (Desmedt and Frankel, 1992; Desmedt and Frankel, 1990). The n leaders generate a shared key, so in order to issue a valid name assertion to define a new member, at least, the signature of k leaders is needed. The generated shared key acts as a virtual leader of the group, it is the key defining the group and sings the assertions. Name assertions are maintained by the leader of the group. This procedure is considerably more complex than the previous ones, but its use will be sporadic since only applications with high security requirements will use it.

In (Ellison and Dohrmann, 2003) the authors present as similar approach to the *set-leader* group, but there, the leaders of the group are not equals in terms of group membership. There is an original leader, which then adds new leaders to the group. In our case, a set of users can agree to set up a group, and all of them will have the same leadership level.

4. Possession paradigm

In order to provide the *poppies* with a management infrastructure, the A-MAPOLA framework relies in a simple *possession paradigm.* A poppy may take control (take possession) of other ones to coordinate a given task. In this case we consider two types of poppies:

Control Station (CS) poppy . A Control Station is a poppy that can control and manage other poppies. It will normally be a strong poppy, which can coordinate several entities to perform a concrete task.

Simple poppy : A simple poppy (or simply, a poppy), is aæ poppy that does not need to control or manage other ones.

An important notion in AMAPOLA, is *holdership* and *ownership.* Each poppy has an *owner* associated to it. The ownership is an static and immutable property of the poppy. It makes reference to the origin of the entity, which will normally be the creator of the specific application or service supplier. The owner of an entity is the ultimate responsible for the entity. If an entity misbehaves or produces some erratic execution due to bugs, the owner could be made responsible for it. The owner has also to take care of the execution of its entities, ensuring that an idle entity does not run forever idle, providing a potential denial of service. This is accomplished by a simple heart-breath protocol, where a CS from the owner may get the status of its entities every given period of time. There may be also third party applications monitoring the networks to detect malfunction and misbehavior such as distributed intrusion detection systems.

Beside the owner, there is the *holder*. Each poppy can have one or several holders, or none if it is idle. A holder is a CS, which is using the entity for an specific application or service and normally for a temporary period of time. The notion of *holder* gives cause for the *possession* paradigm.

Possession protocols

The main idea is to provide protocols as simple as possible, that can be extended and combined to support more complex interactions. This protocols deal mainly with the management of poppies, and more precisely with the *possession* of entities, that is, how to become a *holder* of other entities, and related actions. These protocols are currently defined in SAML over FIPA[2]'s Agent Communication Language and ontologies, although given its simplicity it is easy to use other ontologies or languages. In fact, the last prototype uses SAML protocols over SOAP.

The main possession protocols are:

- *Take-possession*: this protocol allows a CS to become the *holder* of another entity. This is achieved in a two step protocol where both entities interchange their public keys.

- *Terminate-possession*: since the possession of a poppy is ordered and initiated by a CS, in a normal situation, it has to be terminated by the same CS. Only a *holder* of a poppy can ask for a termination of the current possession, and the CS stops being the *holder* of the poppy.

- *Revoke-possession*: there are some situations where the *held* entity may initiate the termination of the possession. This situations does not correspond to the normal operation between the holder and the poppy, thus we refer to them as *revocation* of possession. The revocation can occur

[2] Foundation for Intelligent Physical Agents: `http://www.fipa.org`.

because the entity is detecting a malfunction, has to stop doing its tasks, is going to be stopped (shutdown, killed, ...), or by direct indication of the owner.

- *Delegate-possession*: a CS may delegate the possession of a poppy to another CS. This is very useful in situations where there are complex interactions between several CSs and entities. CSs can exchange their held poppies. This protocol is initiated by a control station CS_1, in possession of a poppy, to delegate it to another CS, CS_2. Then, CS_1 is no longer a *holder* of the poppy, and CS_2 becomes a new *holder*. The poppy cannot deny the delegation, nevertheless, after the delegation, the poppy can revoke the possession of CS_2 if it needs to.

This protocols can be used to handle single poppies or groups of them. To manage groups, the protocol is initiated with one of the group leaders, which is responsible for propagating the protocol to the other members of the group. Given that a CS can be the holder of another CS, possession can also be cascaded through entities. A CS may possess another CS, which in turns possesses another poppy.

Some security considerations

AMAPOLA was designed with security in mind, and the possession protocols are an example. One of the objectives was to provide a practical framework to accommodate several possible solutions. For instances, the possession protocols and principles makes it feasible to accommodate security policy models similar to the *The Resurrecting Duckling*(Stajano and Anderson, 2000; Stajano, 2001). There, a device recognizes as its owner the first entity that sends it a secret key[3]. The process is called *imprinting*. The policy describes several mechanisms to manage this *imprinting*, terminate it and so on. In our case the *imprinting* may be made by taking possession of the entity. One difference with the resurrecting duckling model, is that AMAPOLA allows a poppy to have more than one holder.

An important issue in ad-hoc networks and ubiquitous computing in general is authentication. There are no warranties of having an on-line server that could act as an authority (even in a distributed fashion). Thus, the possession protocols may assume an *anonymous authentication* approach. When a CS wants to take control of a poppy that serves and audio stream, i does it. The CS does not need to know the identity of the poppy, it just needs to know that serves an audio stream and that it can be used. This idea is also used in *trust management* systems such as (Ellison et al., 1999; Blaze et al., 1999), which

[3] By *secret key* we refer to the key of a symmetric cryptogram.

claim that you do not really care who your interlocutor is, so long as she carries the right credentials.

5. Implementation Details

The initial implementation of AMAPOLA is made in Java on top of the *Java Agent DEvelopment Framework* (JADE) (`http://jade.tilab.com/`). JADE is a popular open source multiagent platform, which also has a lightweight version (JADE-LEAP) that can be executed in J2ME (Java 2 Micro Edition).

AMAPOLA is intended to facilitate the development of applications in pervasive networked environments. It mainly consists of a simple API, which is presented to the programmer as services. There are currently three main services:

- *AmapolaIdentity*: provides the identity of the poppy and naming related functionality, including the name container for the poppy.
- *ControlStationPoppy*: provides the functionality for the possession protocols for a CS.
- *SimplePoppy*: provides the functionality for the possession protocols for a simple poppy.

To create a poppy, the programmer just has to include the required service in it main agent class. The way to do it is by composition and delegation, this way it does not interfere with the possible existing inheritance hierarchy of the agent. Thus we favor composition over class inheritance (Gamma et al., 1995). Figure 1 shows a very simplified and schematic organization of the AMAPOLA API from the programmers point of view.

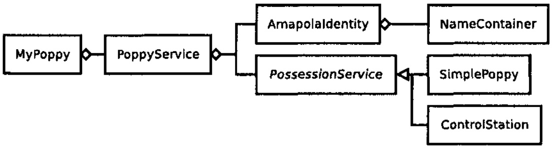


Figure 1. Amapola API outline.

AMAPOLA also provides tools to help in the development and testing of applications. The *CS-console*, presents to the user a graphical interface, which provides the main functionality of a CS so it can be used to test current applications or help in the development of new ones.

6. Conclusions

In this paper we have presented AMAPOLA, a framework for developing applications in ubiquitous computing environments. It provides a simple distributed infrastructure to identify entities (called poppies) and manage groups. It also provides simple protocols to manage the entities. Security is an important issue in AMAPOLA, as well as to easy the task of developers. We have outlined the implementation of the framework, which currently runs on top of a popular multiagent platform (JADE).

The framework makes use of SAML to express the information and the protocols, which makes it easy to interact with other standardized applications in fields such as Web Services, or Grid.

References

3APL-M: Platform for Lightweight Deliberative Agents. `http://www.cs.uu.nl/3apl-m/references.html`.

Blaze, M., Feigenbaum, J., Ioannidis, J., and Keromytis, A. (1999). The KeyNote Trust Management System. RFC 2704, IETF.

Desmedt, Y. and Frankel, Y. (1992). Shared generation of authenticators and signatures. In *CRYPTO '91: Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, pages 457–469. Springer-Verlag.

Desmedt, Yvo and Frankel, Yair (1990). Threshold cryptosystems. In *CRYPTO '89: Proceedings of the 9th Annual International Cryptology Conference on Advances in Cryptology*, pages 307–315. Springer-Verlag.

Ellison, C. and Dohrmann, S. (2003). Public-key support for group collaboration. *ACM Trans. Inf. Syst. Secur.*, 6(4):547–565.

Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., and Ylonen, T. (1999). SPKI Certificate Theory. RFC 2693, IETF.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns. Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional Computing Series.

Helal, S. (2005). Standards & emerging technologies. *IEEE Pervasive Computing*, 4(1).

Helal, S., Mann, W., El-Zabadani, H., King, J., Kaddoura, Y., and Jansen, E. (2005). The gator tech smart house: A programmable pervasive space. *IEEE Computer*, 38(3).

Holmquist, L. E., Gellersen, H. W., Kortuem, G., Antifakos, S., Michahelles, F., Schiele, B., Beigl, M., and Maze, R. (2004). Building intelligent environments with smart-its. *IEEE Computer Graphics and Applications*, 24(1).

S. Cantor, J. Kemp, R. Philpott and E. Maler, ed. (2005). Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. OASIS XACML-TC, Committee Draft 04.

Sloman, M. (2001). Will pervasive computers be manageable? Keynote Talk HP OpenView 2001, New Orleans.

Stajano, F. (2001). The resurrecting duckling - what next? In *Revised Papers from the 8th International Workshop on Security Protocols*, pages 204–214. Springer-Verlag.

Stajano, F. and Anderson, R. J. (2000). The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Proceedings of the 7th International Workshop on Security Protocols*, pages 172–194.

DECOUPLING COMPONENTS OF AN ATTACK PREVENTION SYSTEM USING PUBLISH/SUBSCRIBE*

Joaquín García[1], Michael A. Jaeger[2], Gero Mühl[2], and Joan Borrell[1]

[1] *Autonomous University of Barcelona,*
Dept. of Information and Communications Engineering,
Edifici Q, 08193 Bellaterra, Spain
{jgarcia,jborrell}@ccd.uab.es

[2] *Technical University of Berlin,*
Institute for Telecommunication Systems,
Communication and Operating Systems Group,
EN6, Einsteinufer 17, D-10587 Berlin, Germany
{michael.jaeger,g_muehl}@acm.org

Abstract Distributed and coordinated attacks can disrupt electronic commerce applications and cause large revenue losses. The prevention of these attacks is not possible by just considering information from isolated sources of the network. A global view of the whole system is necessary to react against the different actions of such an attack. We are currently working on a decentralized attack prevention framework that is targeted at detecting as well as reacting to these attacks. The cooperation between the different entities of this system has been efficiently solved through the use of a publish/subscribe model. In this paper we first present the advantages and convenience in using this communication paradigm for a general decentralized attack prevention framework. Then, we present the design for our specific approach. Finally, we shortly discuss our implementation based on a freely available publish/subscribe message oriented middleware.

1. Introduction

When attackers gain access to a corporate network by compromising authorized users, computers, or applications, the network and its resources can be-

*This work has partially been funded by the Spanish Ministry of Science and Technology (MCYT) through the project TIC2003-02041 and the Catalan Ministry of Universities, Research and Information Society (DURSI) with its grant 2003FI-126.

come an active part of a globally distributed or coordinated attack. Such an attack might be a coordinated port scan or distributed denial of service attack against third party networks — or even against computers on the same network. Both, distributed and coordinated attacks, rely on the combination of actions performed by a malicious adversary to violate the security policy of a target computer system. To prevent these attacks, a global view of the system as a whole is necessary. Hence, different events and specific information must be gathered and combined from all the sources. This affects, for example, information about suspicious connections, initiation of processes, and addition of new files.

We are currently working on the design and development of an attack prevention framework that is targeted at detecting as well as reacting to distributed and coordinated attack scenarios [García et al., 2004]. Our approach is based on gathering and correlating information held by multiple sources. We use a decentralized scheme based on message passing to share alerts in a secure communication infrastructure. This way, we can detect and prevent these kind of attacks performing detection and reaction processes based on the knowledge gained through alert correlation.

In this paper we propose a decentralized infrastructure to share alerts between components. The information exchange between peers is intended to achieve a more complete view of the system in whole. Once achieved, one can detect and react on the different actions of a coordinated or distributed attack.

The rest of this paper is organized as follows: We start with an introduction to the publish/subscribe communication paradigm in Section 2 where we present the advantages and convenience in using this model for our problem domain and analyze related work. In Section 3, we discuss the communication mechanism used to exchange information among the components of our system using xmlBlaster, an open source publish/subscribe message oriented middleware [Ruff, 2000] and present the current state of our implementation. We close with conclusions and give an outlook on future work in Section 4.

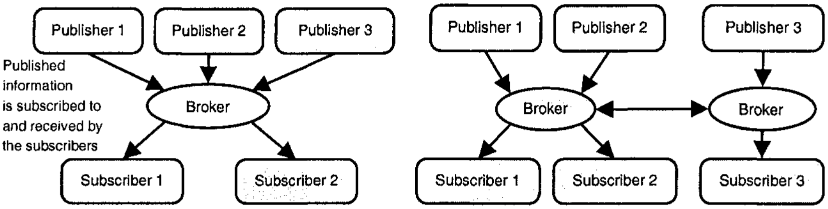
2. Publish/Subscribe Model

The publish/subscribe communication model is intended for group communication, i.e. for situations where a message (*notifications*) sent by a single entity is required by, and should be distributed to, multiple entities. It is often used for efficient and comfortable information dissemination to group members which may have individual interests in arbitrary subsets of messages published. In contrast to multicast communication, clients have the possibility to describe the events they are interested in more precisely (e.g. based on the contents of the notification). Clients can choose to either subscribe or unsubscribe to messages as time goes by, and all the subscribers are independent of each other.

Publish/Subscribe Systems

A publish/subscribe system consists of at least one broker forwarding notifications published by clients to other clients that are interested in them. For scalability reasons, it is common to implement a distributed broker network that forms a so-called *notification service* through an overlay network consisting of brokers. This service provides a distributed infrastructure for notification routing which includes the management of subscriptions and the dissemination of notifications in a possibly asynchronous way. Clients can publish notifications and subscribe to filters that are matched against the notifications passing through the broker network. If a broker receives a new notification it checks if there is a local client that has subscribed to a filter that matches this notification. If so, the message is delivered to this client. Additionally, the broker forwards the message to neighbor brokers according to the applied routing algorithm. We refer to [Mühl, 2002] for a good survey on the field.

An example of a simple centralized publish/subscribe system is shown in Figure 1(a). Here, five clients are connected to a single broker: three clients that are publishing notifications and two clients that are subscribed to a subset of the notifications published on the broker. Subscribers can choose to subscribe to the notifications available through the broker or cancel existing subscriptions as needed. The broker matches the notifications it received from the publishers to the subscriptions, ensuring this way that every publication is delivered to all interested subscribers.



(a) Simple publish/subscribe system.

(b) Extended pub/sub system.

Figure 1. Examples for publish/subscribe environments.

This very basic publish/subscribe setup can be extended by connecting multiple brokers (cf. Figure 1(b)), enabling them to exchange messages. The extended design allows subscribers on one of the brokers to receive messages that have been published on another broker, further freeing the subscriber from the constraints of connecting to the same broker the publisher is connected to. Most available implementation make this transparent for the programmer by keeping the same interface operations as in the centralized design. This way, an application can easily be distributed. The subscribers are able to formulate their interests based e.g. on the contents of the notifications and a special attribute they carry. This is known as content-based and topic-based subscription, respectively.

Topic-based subscriptions are easier to handle than content-based subscriptions. Subscribers specify their interest in a topic and receive all messages published on this topic. Two different matching mechanisms are commonly used here. One matches subscriptions successfully to notifications if the topic of the subscription exactly matches the topic under which the notification is published. Using this mechanism, topics become equivalent to "channels". The other mechanism arranges topics in a subject tree such that subscriptions not only match notifications if the topics are the same, but also if the topic of the subscription is an ancestor of the notification topic in the subject tree (in this case, a topic becomes equivalent to a "theme").

Content-based subscriptions allow more sophisticated subscriptions on the cost of higher matching load and more complex routing decisions. Here, a subscription can be formulated extremely fine-grained based on the content of notifications using a query language that can be arbitrarily complex. Moreover, there does not have to be a system wide agreement on the set of topics as it is generally a good idea for topic based routing.

Related Work

Traditional client/server solutions for the prevention of distributed and coordinated attacks can quickly become a bottleneck due to saturation problems associated with the service offered by centralized or master domain analyzers. A master domain analyzer is the entity on top of a hierarchy of IDSs consisting of multiple analyzers and different domains to analyze. Centralized systems, such as DIDS [Snapp et al., 1991] and NADIR [Hochberg et al., 1993], use this approach to process their data in a central node although the collection of data is distributed. These schemes are straightforward as they simply push the data to a central node and perform the computation there. Hierarchical approaches, such as GrIDS [Staniford-Chen et al., 1996] and NetSTAT [Vigna and Kemmerer, 1999], have a layered structure where data is locally preprocessed and filtered. Although they mitigate some weaknesses present in centralized schemes, they still cannot avoid bottlenecks, scalability problems, and fault tolerance issues due to vulnerabilities at the root level.

In contrast to these traditional designs, alternative approaches try to eliminate the need for dedicated elements. The idea of distributing the detection process has some advantages regarding centralized and hierarchical approaches. Mainly, decentralized architectures have no single point of failure and bottlenecks can be avoided. Some message passing designs, such as CSM [White et al., 1999] and Quicksand [Kruegel, 2002], try to eliminate the need for dedicated elements by introducing a peer-to-peer architecture. Instead of having a central monitoring station to which all data has to be forwarded, there are independent uniform working entities at each host performing similar basic operations. To detect coordinated and distributed attacks, the different entities have to collaborate on the detection activities and cooperate to perform a decentralized correlation algorithm.

These designs seem to be a promising technology to implement decentralized architectures for the detection of attacks. However, the presented systems still exhibit very simplistic designs and suffer from several limitations. For instance, in some of them, every node has to have complete knowledge of the system: All nodes have to be connected to each other which can make the matrix of the connections, that are used for providing the alert exchanging service, grow explosively and become very costly to control and maintain. Another important disadvantage present in this design is that the different entities always need to know where a received notification has to be forwarded (similar to a queue manager). This way, when the number of possible destinations grows, the network view can become extremely complex, which leads to a system that is not scalable. Other designs are based on flooding which makes the system easier to maintain on the cost of scalability, as the message complexity grows fast with the number of brokers.

Most of these limitations can be solved efficiently by using a publish/subscribe based system. The advantage of this model for our problem domain over other communication paradigms is on the one hand that it keeps the producer of messages separated from the consumer and on the other hand that the communication is information-driven. This way, it can avoid problems regarding the scalability and the management inherent to other designs, by means of a network of publishers, brokers, and subscribers. A publisher in a publish/subscribe system does not need to have any knowledge about any of the entities that consume the published information. Likewise, the subscribers do not need to know anything about the publishers. New services can simply be added without any impact on or interruption of the service to other users.

3. Alert Communication Infrastructure

This section describes the alert communication infrastructure and implementational details of our approach. As our motivation is not targeted on developing a new publish/subscribe system, we try to reuse as much available code and tools as possible. For our experiments we used *xmlBlaster*, an open source publish/subscribe message oriented middleware [Ruff, 2000]. It connects a set of nodes that build up the infrastructure for exchanging alerts using the interface operations offered by the underlying middleware.

Alerts are formulated using XML as this is the standard format in xmlBlaster. Each message consists of a header filtering can be applied to, a body, and a system control section. Filters are XPath expressions that are evaluated over the header to decide if a message has to be delivered to a subscriber. We discuss the essential interface operations offered by xmlBlaster in the following section.

Interface Operations

Conceptually, the alert communication infrastructure offered through xmlBlaster can be viewed as a black box with an *interface.* It offers a number of *operations*, each of which may take a number of *parameters*. Clients can invoke *input operations* from the outside, and the system itself invokes *output operations* to deliver information to clients. We list the main operations that are of interest for our work in Figure 2.

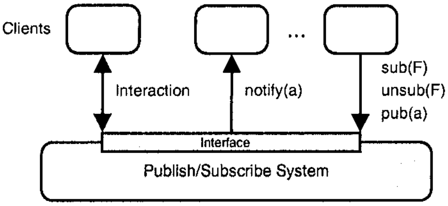


Figure 2. Black box view of a publish/subscribe system.

To publish alerts, clients invoke the *pub*(a) operation, giving the alert a as parameter. The published alert can potentially be delivered to all clients connected to the system via an output operation called *notify*(a). Clients register their interest in specific kinds of alerts by issuing subscriptions via the *sub*(F) operation, which takes a filter F as parameter. Each client can have multiple active subscriptions which must be revoked separately by using the *unsub*() operation.

All these operations are instantaneous and take parameters from the set of all clients $\mathcal{C}$, the set of all alerts $\mathcal{A}$, and the set of all filters $\mathcal{F}$. Formally, a filter $F \in \mathcal{F}$ is a mapping defined by

$$F: \quad a \longrightarrow \{\text{true}, \text{false}\} \qquad \forall a \in \mathcal{A}$$

We say that a *notification* n *matches filter* $F \in \mathcal{F}$ iff $F(a) = \text{true}$. We also assume that each alert can only be published once and that every filter is associated with a unique identifier in order to enable the alert communication infrastructure to identify a specific subscription.

Components and Interactions

As shown in Figure 3, each node of the architecture is made up of a set of local analyzers (with their respective detection units or sensors), a set of alert managers (to perform alert processing and manipulation functions), and a set of local reaction units (or effectors). These components, the interactions between them, and the alert communication infrastructure, are described below.

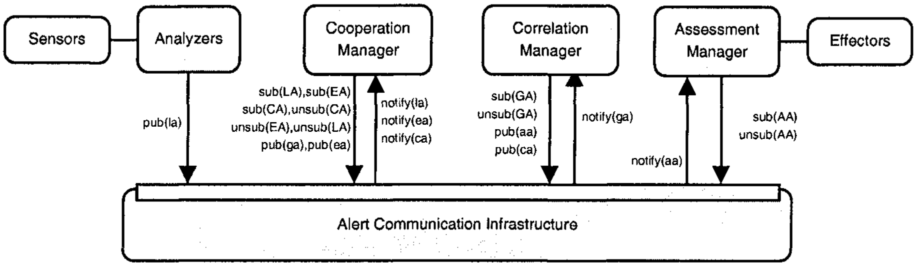


Figure 3. Overview of the Attack Prevention Framework.

Analyzers. Local elements which are responsible for processing local audit data are called *analyzers*. They process the information gathered by associated sensors to infer possible alerts. Their task is to identify occurrences which are relevant for the execution of the different steps of an attack and pass this information to the correlation manager via the publish/subscribe system. They are interested in local alerts. Each local alert is detected in a sensor's input stream and published through the publish/subscribe system by invoking the *pub*(*la*) operation, giving the *local alert la* as parameter.

Each notification *la* has a unique classification and a list of attributes with their respective types to identify the analyzer that originated the alert (*AnalyzerID*), the time the alert was created (*CreateTime*), the time the event(s) leading up to the alert was detected in the sensor's input stream (*DetectTime*), the current time on the analyzer (*AnalyzerTime*), and the source(s) and target(s) of the event(s) (*Source* and *Target*). All possible classifications and their respective attributes must be known by all system components (i.e. sensors, analyzers and managers) and all analyzers are capable of publishing instances of local alerts of arbitrary types.

Local alerts are exchanged using IDMEF messages [Debar et al., 2005]. The *Intrusion Detection Message Exchange Format* (IDMEF) is proposed as a standard data format for automated intrusion detection systems to raise alerts about events they report as suspicious. It allows analyzers and managers to assemble very complex alert descriptions.

Managers. Performing aggregation and correlation of local alerts and external events is the task of *managers*. While using multiple analyzers and sensors together with heterogeneous detection techniques increases the detection rate, it also increases the number of alerts to process. In order to reduce the number of false negatives and distribute the load that is imposed by the alerts our architecture provides a set of cooperation and correlation *managers*, which perform aggregation and correlation of both, local alerts (i.e., messages provided by the node's analyzers) and external messages (i.e., the information received from other collaborating nodes).

Cooperation Managers. The basic functionality of each cooperation manager is to cluster alerts that correspond to the same occurrence of an action. Each cooperation manager registers its interest in a subset $\mathcal{L}_A$ of local alerts published by analyzers on the same node by invoking the *sub*(LA) operation, which takes the filter LA as parameter, with

$$LA(a) = \begin{cases} \text{true} & , \quad a \in \mathcal{L}_A \\ \text{false} & , \quad \text{otherwise.} \end{cases}$$

Similarly, the cooperation manager also registers its interest in a set of related external alerts $\mathcal{E}_A$ by invoking the *sub*(EA) operation with filter EA as parameter, and

$$EA(a) = \begin{cases} \text{true} & , \quad a \in \mathcal{E}_a \\ \text{false} & , \quad \text{otherwise.} \end{cases}$$

Finally, it registers its interest in local correlated alerts $\mathcal{C}_A$ by invoking the *sub*(CA) operation with

$$CA(a) = \begin{cases} \text{true} & , \quad a \in \mathcal{C}_A \\ \text{false} & , \quad \text{otherwise.} \end{cases}$$

Once subscribed to these three filters, the alert infrastructure will notify the subscribed managers of all matching alerts via the output operations *notify*(la), *notify*(ea) and *notify*(ca) with $la \in \mathcal{L}_A$, $ea \in \mathcal{E}_A$ and $ca \in \mathcal{C}_A$. All notified alerts are processed and, depending on the clustering and synchronization mechanism, the cooperation manager can publish global and external alerts by invoking *pub*(ga) and *pub*(ea). Finally, it can revoke active subscriptions separately by using the operations *unsub*(CA), *unsub*(EA) and *unsub*(LA).

Correlation Managers. The main task of this manager is the correlation of alerts described in [García et al., 2004]. It operates on the set of global alerts $\mathcal{G}_A$ published by the local cooperation manager. To register its interest in these alerts, it invokes *sub*(GA), which takes the filter GA as parameter with

$$GA(a) = \begin{cases} \text{true} & , \quad a \in \mathcal{G}_A \\ \text{false} & , \quad \text{otherwise.} \end{cases}$$

The alert infrastructure will then notify the correlation manager of all matched alerts with the output operation *notify*(ga), $ga \in \mathcal{G}_A$. Each time a new alert is received, the correlation mechanism finds a set of action models that can be correlated in order to form a scenario leading to an objective. Finally, it includes this information into the *CorrelationAlert* field of a new IDMEF message and publishes the correlated alert by invoking *pub*(ca), giving the notification $ca \in \mathcal{C}_A$ as parameter. To revoke the subscription, it uses *unsub*(GA).

The correlation manager is also responsible for reacting on detected security violations. The algorithm used is based on the anti-correlation of actions to select appropriate countermeasures in order to react and prevent the execution of the whole scenario [García et al., 2004]. As soon as a scenario is identified, the correlation mechanism looks for possible action models that can be anti-correlated with the individual actions of the supposed scenario, or even with the goal objective. The set of anti-correlated actions represents the set of countermeasures available for the observed scenario. The definition of each anti-correlated action contains a description of the countermeasures which should be invoked (e.g. hardening the security policy). Such countermeasures are included into the *Assessment* field of a new IDMEF message and published by invoking *pub*(aa), using the *assessment alert* aa as parameter.

Assessment Managers. Another manager called *assessment manager* will register and revoke its interest in these assessment alerts by invoking *sub*(AA) and *unsub*(AA). Once notified, the assessment manager performs post-processing of the received alerts before sending the corresponding reaction to the local response units.

Implementation

We deployed a set of three analyzers publishing ten thousand messages to evaluate our implementation of the alert communication infrastructure for the proposed architecture. Therefore, we used the *DARPA Intrusion Detection Evaluation Data Sets* [Lippmann et al., 2000] where more than 300 instances of 38 different automated attacks were launched against victim hosts in seven weeks of training data and two weeks of test data. These messages were published as local alerts through the communication infrastructure, and then processed and republished in turn to three subscribed managers. The evaluation on the alert communication infrastructure proved to be satisfactory, obtaining a throughput performance higher than 150 messages per second on an Intel-Pentium M 1.4 GHz processor with 512 MB RAM, analyzers and managers on the same machine running Linux 2.6.8, using Java HotSpot Client VM 1.4.2 for the Java based broker. Message delivery did not become a bottleneck as all messages were processed in time and we never reached the saturation point. This result gives us good hope that using a publish/subscribe system for the communication infrastructure indeed increases the scalability of the proposed architecture.

The implementation of both analyzers and managers was based on the *libidmef* C library [Migus, 2004] which was used to build and parse compliant IDMEF messages. The communication between analyzers and managers through xmlBlaster brokers was based on the xmlBlaster internal socket protocol and implemented using the xmlBlaster client C socket library [Ruff, 2000], which provides asynchronous callbacks to Java based brokers. The managers formulated their subscriptions using XPath expressions, filtering the messages they wished to receive from the broker.

4. Conclusions

We presented an infrastructure to share alerts between the components of a prevention framework. The framework itself is targeted at detecting as well as reacting to distributed and coordinated attack scenarios through the use of the publish/subscribe communication paradigm. In contrast to traditional client/server solutions, where centralized or hierarchical approaches quickly become a bottleneck due to saturation problems associated with the service offered by centralized or master domain analyzers, the information exchange between peers in our design achieves a more complete view of the system in whole. We believe that this is necessary to detect and react on the different actions of an attack. We also introduced an implementation based on an open source publish/subscribe message oriented middleware and conducted experiments showing that the architecture is performant enough for the application in real-world scenarios.

As future work we are considering to secure the communication partners by utilizing the SSL plugin for xmlBlaster. This way, each collaborating node will receive a private and a public key. The public key of each node will be signed by a certification authority (CA), that is responsible for the protected network. Hence, the public key of the CA has to be distributed to every node as well. The secure SSL channel will allow the communicating peers to communicate privately and to authenticate each other, thus preventing malicious nodes from impersonating legal ones. The implications coming up with this new feature, such as compromised key management or certificate revocation, will be part of this work. We are also planning a more in-depth study about privacy mechanisms by exchanging alerts in a pseudonymous manner. By doing this, we hope that we can provide the destination and origin information of alerts (*Source* and *Target* field of IDMEF messages) without violating the privacy of publishers and subscribers located on different domains. Our study will cover the design of a pseudonymous identification scheme, trying to find a balance between identification and privacy.

Acknowledgments

The collaboration between J. García, F. Cuppens, and F. Autrel sharpened many of the arguments presented in this paper.

The work of J. García and J. Borrell is funded by the Spanish Government CICYT Commission and the Catalan Government DURSI Ministry. Michael A. Jaeger is funded by Deutsche Telekom Stiftung. Gero Mühl is funded by Deutsche Telekom.

References

[Debar et al., 2005] Debar, H., Curry, D., and Feinstein, B. (January 2005). Intrusion detection message exchange format data model and extensible markup language. Technical report.

[García et al., 2004] García, J., Autrel, F., Borrell, J., Castillo, S., Cuppens, F., and Navarro, G. (2004). Decentralized publish-subscribe system to prevent coordinated attacks via alert correlation. In *Sixth International Conference on Information and Communications Security*, volume 3269 of *LNCS*, pages 223–235, Málaga, Spain. Springer-Verlag.

[Hochberg et al., 1993] Hochberg, J., Jackson, K., Stallins, C., McClary, J. F., DuBois, D., and Ford, J. (May 1993). NADIR: An automated system for detecting network intrusion and misuse. In *Computer and Security*, volume 12(3), pages 235–248.

[Kruegel, 2002] Kruegel, C. (June 2002). *Network Alertness - Towards an adaptive, collaborating Intrusion Detection System.* PhD thesis, Technical University of Vienna.

[Lippmann et al., 2000] Lippmann, R., Haines, J., Fried, D., Korba, J., and Das, K. (2000). The 1999 darpa off-line intrusion detection evaluation. *Computer Networks*, (34):579–595.

[Migus, 2004] Migus, A. C. (March 2004). IDMEF XML library version 0.7.3. `http://sourceforge.net/projects/libidmef/`.

[Mühl, 2002] Mühl, G. (2002). *Large-Scale Content-Based Publish-Subscribe Systems.* PhD thesis, Technical University of Darmstadt.

[Ruff, 2000] Ruff, M. (2000). XmlBlaster: open source message oriented middleware. `http://xmlblaster.org/`.

[Snapp et al., 1991] Snapp, S. R., Brentano, J., Dias, G. V., Goan, T. L., Heberlein, L. T., Ho, C., K. N. Levitt, Mukherjee, B., Smaha, S. E., Grance, T., Teal, D. M., and Mansur, D. (October, 1991). DIDS (distributed intrusion detection system) - motivation, architecture and an early prototype. In *Proceedings 14th National Security Conference*, pages 167–176.

[Staniford-Chen et al., 1996] Staniford-Chen, S., Cheung, S., Crawford, R., Dilger, M., Frank, J., Levitt, J. Hoagland K., Wee, C., Yip, R., and Zerkle, D. (1996). GrIDS – a graph-based intrusion detection system for large networks. In *Proceedings of the 19th National Information Systems Security Conference.*

[Vigna and Kemmerer, 1999] Vigna, G. and Kemmerer, R. A. (1999). NetSTAT: A network-based intrusion detection system. *Journal of Computer Security*, 7(1):37–71.

[White et al., 1999] White, G. B., Fisch, E. A., and Pooch, U. W. (February 1999). Cooperating security managers: A peer-based intrusion detection system. *IEEE Network*, 7:20–23.

A P2P COLLABORATIVE FRAMEWORK FOR SMARTPHONE†

Heien-Kun Chiang[1], Feng-Lan Kuo[2], Meng-Ting Chen[1]

[1] Information Management Department, National Changhua University of Education
hkchiang@cc.ncue.edu.tw
[2] English Department & Graduate Institute for Children's English, National Changhua University of Education
laflkuo@cc.ncue.edu.tw

Abstract. This study investigates the incorporation of peer-to-peer (P2P) computing model into smartphone, and discusses its potential benefits for collaborative mobile users. The goal of this study is to establish a P2P computing environment for network-capable smartphone. Besides, in order to support the interoperability of smartphones and allow them to share information seamlessly, this study proposes a service oriented P2P framework, utilizing XML and SOAP as the underlying communication media. A prototyping system of the proposed system is implemented. Its implications and applications are also discussed.

1 Introduction

Smartphone is a mobile device with the voice communication capability of cellular phone and the PIM (Personal Information Management) functionality of PDA. Its slim foam factor enables it to be carried around easily and its computing power and PIM capability allow businessmen to continue their work while on the run. The proliferation of smartphone is a crucial component of the anticipated development of pervasive computing. Experts predict that for smartphones to emerge into the pervasive computing environment they must be ease of use and must provide practical solutions to people's everyday problems [1].The wireless communication technology has also advanced at an unprecedented speed. The bandwidth of WLAN has gone up from 802.11b 11M bps to 802.11a 55M bps and many PDAs have built-in 802.11b WLAN network card. Furthermore, from GSM to GPRS, and now WCDMA, the bandwidth of cellular network has risen up from 9.6 kbps to 2M bps. Smartphones, with right networking device and access points of Bluetooth, WLAN, and WCDMA networks, are able to connect to the network and interact with other computing devices anytime, anyplace. This creates a dynamic, decentralized, ubiquitous environment where smartphones are able to share services with nearby devices after they have been discovered and they have established relationships with smartphones [2].

† This work is sponsored by National Science Council, Taiwan (NSC93-2515-S-018-006 & NSC94-2520-S-018-002).

The goal of pervasive computing environment is to provide a dynamic, active space where a spectrum of smartphones can freely communicate with each other and share resources. Through the seamless integration of heterogeneous smartphones, new functionality can be created, user productivity can be enhanced, human thought and activity with digital information can be augmented, and the overall daily tasks can be simplified [3]. In short, pervasive computing aims to unobtrusively assist work or recreational activities with information technology that optimizes the environment for people's needs no matter they are at home, office, or public spaces [4]. The current state of the art of pervasive computing is still in its infant stage, facing issues of providing effective supports to enhance the resources sharing and collaborative work for smartphones.

In addition to the advances on mobile devices and wireless networks, the increase in computing power and storage capability of computers has offered peer-to-peer distributed model a new application area. The widespread use of Napster and Gnutella [5] has caused Peer-to-Peer (P2P) computing model to gain the attention of many researchers and companies alike. Most P2P researches focus on how to effectively lookup and utilize the resources in wired network or how to increase the performance of P2P model. However, issues regarding the interoperability of smartphones, the acquiring and sharing of services after being discovered, and the support of collaborative work after being connected in mobile network are not extensively researched.

Fortunately, the emergence of Web services comes to rescue. Web services have been proposed by the W3C as the standard mechanisms for web applications to communicate and exchange information. The power of Web services comes from the interoperability and extensibility of XML. Through the standard programmatic interfaces, Web services can be combined in a loosely coupled way to achieve complex operations. Sophisticated value-added services can be delivered through the interaction of programs providing Web services [6,7]. This paper proposes a Web services based P2P collaborative framework to support the development of collaborative mobile applications for smarphones. By examining the architectural models of Web services and P2P, and the structural requirements of collaborative mobile applications, the proposed framework provides collaborative Web services and P2P facilities to allow smartphones to connect to each other dynamically and to share information seamlessly.

2 Smartphones, P2P, and Web Services Model

In post PC (Personal Computer) era, the trend of computing device moves towards small form-factor, light, and portable devices with faster processing speed and network connection ability. This is evident from the widespread use of diverse PDAs and lately smartphones. Figure 1 shows some of the most popular commercial smartphones and their features.

Model:	Palm Treo600	SonyEricsson P900	Motorola MPx
OS:	Plam OS 5.2	Symbian EPOC 7.0	Mobile Windows
CPU:	ARM 144 MHz	ARM 156 MHz	ARM 200 MHz
Mem:	32M + SD/MMC	13M + SD	32M + SD/MMC
Screen:	320 x 320 pixels	208 x 320 pixels	240 x 320 pixels
Net:	CDMA,GRPS, Bluetooth	CDMA, GRPS, Bluetooth	CDMA, GRPS, Bluetooth, WiFi
MM:	EMS, MMS	EMS, MMS	EMS, MMS

Fig. 1. Comparisons of three popular smartphones

2.1 Smartphones

The characterisitcs of smartphones making them ideal devices for collaborative mobile applications are:

a. Network-ready: Most smartphones have at least one data communication chipset such as Bluetooth, GPRS or WCDMA.
b. Mobility: With wireless network and smartphones, we are moving from an infrastructure of tethered computing and communications toward mobility. That is, smartphones enable us to work even when we are on the move.
c. Multimedia: Smartphones are capable of recording/playing audio/video and playing mobile games in addition to the MMS.
d. Virtuality: The software environment brought by smartphones is similar in functionality to their desktop counterpart to allow people to work whenever they desire and wherever they are.

2.2 P2P Computing Model

P2P was said to be the third network revolution since the invention of Mosaic Web browser. P2P is a communication model in which each party has the same capabilities and either party can initiate a communication session. In most cases, a node in P2P model can act both as a client and as a server depending on whether it acts as a requester or a servant.

P2P can be implemented in hardware as the IBM's APPN (Advanced Peer-to-Peer Networking) that supporting the P2P communication model. Or it can be implemented as a software solution as the likes of Napster where a group of computer users with the same networking program connect with each other and directly access files from one another's hard drives. Contrasting to the P2P is the client/server model where a client makes a service request from a server which fulfills the request of a client. Typically, the server's computing power is far superior to the client.

2.3 Web Services Model

Nowadays, as more and more companies seek to conduct business over the Internet, the problem they face is how to make their applications work with those of their customers and suppliers. With proprietary message format, companies can get their applications to talk to each other. However, as the number of applications goes up, the number of possible communication paths increase dramatically. One solution for this problem is to use middleware as the communication mediator but it does not scale well to large applications.

Web services are applications that adopt a universal language to send data and instructions over the Internet to one another. The goal of Web services is to provide a flexible and cost-effective solution for enterprises conducting business on the Internet. Web services are open, distributed software components, and are built on top of the standards of XML, SOAP, WSDL (Web Services Description Language), and UDDI (Universal Description Discovery and Integration) [8,9]. Programmers can use their favorite programming language and operating system to describe and write Web services which can be invoked over the Web. Please see [10] for detailed descriptions of Web services.

3 The Framework and Implementation

To achieve the interoperability of mobile devices, this study proposes the Web services based service framework utilizing SOAP as the underlying message format to communicate with mobile devices of different platforms.

Figure 2 shows the conceptual diagram of the proposed framework. Through the common interface of Web services module and the P2P mechanism, every mobile device can access the Web services offered by any locatable mobile device. This framework is particularly suitable for dynamic, ad hoc wireless network. But how does the Web services and P2P work in this environment? How does one mobile device know the status of the other in P2P way? To solve these problems, we design mechanisms which allow each mobile device to register its services. After a service provider has written and deployed its services, it registers them to the UDDI registry server, implemented as a SOAP server. Then, a service requester queries the SOAP directory server to see if there exists any service meeting its requirement. If there exists one, the SOAP server forwards the service provider's WSDL to the service requester to allow them to communicate with each other directly without the need of the SOAP server afterwards. This enables the P2P communication between the service requester and the service provider. Figure 3 shows the P2P communication process between two mobile devices. Notice that every mobile device can act as a client (service requester) or a server (service provider) depending on its role.

The proposed framework has the following characteristics:

- Web services: The services offered by a mobile device are encapsulated into Web services, facilitating a uniform communication path among peer mobile devices.
- SOAP-based message: The delivery of message is based on SOAP. Thus, applications written in different programming languages and run on different operating systems can still communicate with each other to achieve the interoperability.
- Service transparency: The location of a service provider is of no concern to a service requester since the binding between them occurs automatically. The selection of a service provider is done in server side and its selection criteria might

be distance or time, depending on the algorithm used by the server.

- Peer-to-Peer: Every mobile device is treated equally as a peer. Everyone has the ability to provide services for others and to use the services from others. This is important for mobile devices because (1) resources in every mobile device are scarce, and (2) performance of mobile devices can be maximized while the load of the server can be minimized.
- Ubiquitous service: Services might not exist when requested but the requester will be notified whenever they appear. This is achieved through UDDI. It eliminates the need of the service requester to continuously poll the server for the service.
- Extensible: The functionalities of applications can be extended by just adding new Web services modules and registering them onto the UDDI server. The programming effort is reduced since the database connection and P2P communication complexity are encapsulated in Web services modules, hidden from applications.

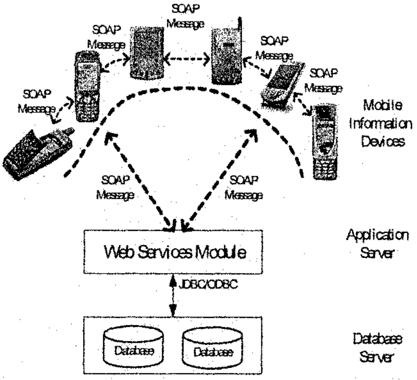


Fig. 2. The conceptual diagram of the proposed framework

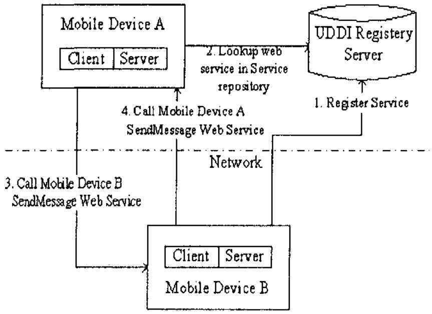

Mobile Device A
Client
Server
2. Lookup web service in Service repository
UDDI Registery Server
1. Register Service
4. Call Mobile Device A SendMessage Web Service
Network
3. Call Mobile Device B SendMessage Web Service
Client
Server
Mobile Device B

Fig. 3. The P2P communication process between two mobile devices

3.1 Layered Design

The design of the proposed framework is based on layered architecture, as shown in Figure 4. In this approach, each layer supports certain functionality which is used by its immediate upper layer.

The proposed framework consists of three layers. The system library layer consists of parsers for parsing XML, SOAP, WSDL, and UDDI messages. The P2P service layer is designed to facilitate the registration, indexing, advertisement, discovery, and notification of services of mobile devices. Collaborative mobile services and multimedia mobile services belong to the application services offered by mobile devices and are encapsulated into Web services. Examples of collaborative mobile services include file sharing, scheduling, instant messaging, and task arrangement. Multimedia mobile services include audio/video messaging, graphics drawing, and gaming.

The advantages of this approach include:

- Simplified design: By dividing functions to each layer, it is much easier to design and implement the framework since one only has to worry about implementing some functionalities for each layer. Besides, flaws are easier to track.
- Adaptability: If a better algorithm is going to replace an old one in a certain layer, other layers will not be affected as long as interfaces between them stay the same.

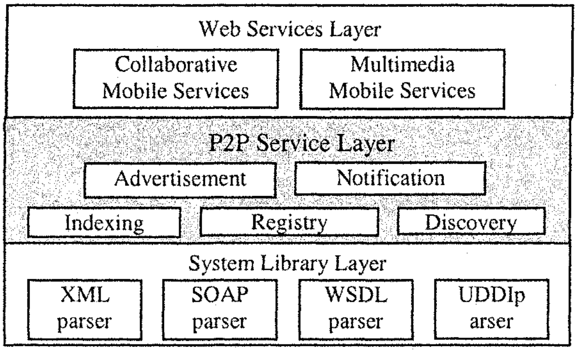


Fig. 4. Layered design of the proposed framework

3.2 Class Diagram

Figure 5 shows the proposed framework's top level class diagram which is composed of seven classes: MobIntMsg (Mobile Instant Messaging), SrvSptWS (Service Support Web Services), MobInfoShr (Mobile Information Sharing), MobGrpInfoMgt (Mobile Group Information Management), ContactWS (Contacts Web Service), TaskWS (Tasks Web Service), and ApptWS (Appointments Web Service)[‡].

[‡] Due to space limit, other classes are omitted here.

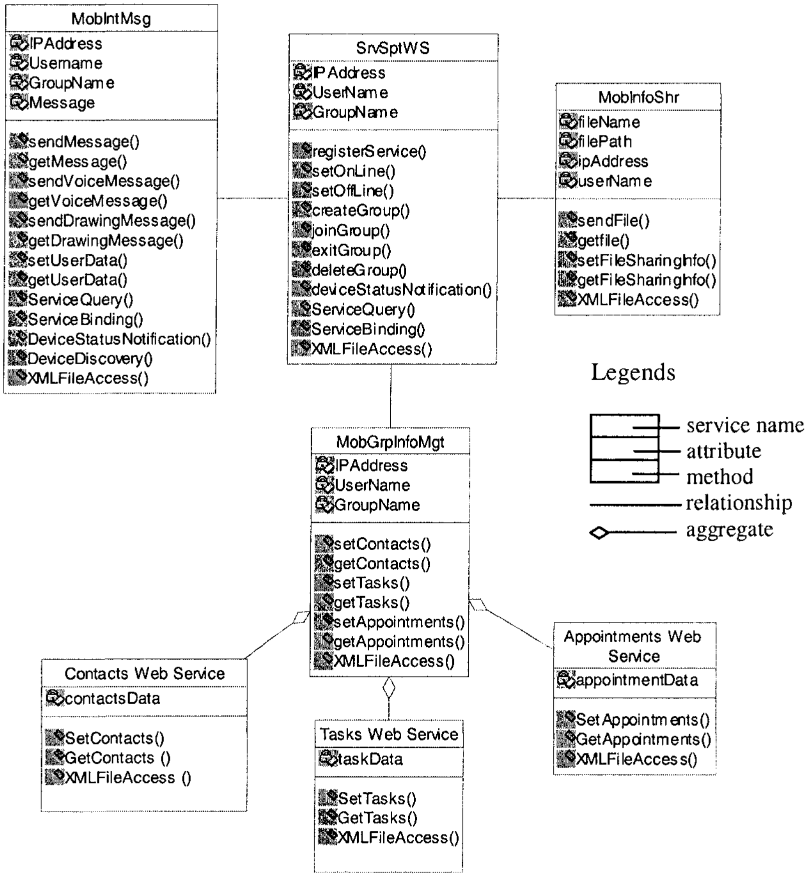


Fig. 5. Class diagram of Web services based P2P collaborative framework

3.3 Implementation

The proposed framework was implemented using Java programming languages. Two personal computers are setup as the UDDI register application server and SOAP application server. The UDDI register application server was implemented using IBM's UDDI service registry and the SOAP application server was built on top of Apache SOAP server. All Web services were implemented using Java and Sun Microsystems's J2EE SDK. Three Compaq iPAQ H3870 with D-Link 802.11b WLAN were used as test beds for smartphones.

3.3.1 XML Configuration and SOAP Message Format

In order for the Web services based P2P collaborative framework for smartphones to work, there must be a mechanism for smartphones to know each other, to understand the capability of each other, and to communicate with each other. All devices related information is stored in a configuration file. Figure 6 shows an XML example of the status information of an iPAQ 3870 PDA with WLAN card. Other configuration files (not shown here) include user information, capability information, directory information, resource information, etc.

```
<?xml version="1.0"?>
<DeviceInfo>                                                   <!-- device infor. -->
   <DeviceName>iPAQ 3870</DeviceName>                          <!-- model -->
   <OSInfo>PocketPC 2002</OSInfo>                              <!-- OS -->
   <DisplayCapability>64K  Colors</ DisplayCapability          <!-- screen -->
>                                                              <!-- CPU -->
   <CPUInfo>StrongARM</CPUInfo>                                <!-- memory-->
   <StoreageSpace>64MB</StoreageSpace>                         <!-- computing power -->
   <ComputeCapability>206Mhz</ComputeCapability>               <!-- electric power -->
   <ElectricPower>1400mAh</ElectricPower >                     <!—network type -->
   <Connection>WLAN</Connection>                               <!-- bandwidth -->
   <Bandwidth>11Mbps</Bandwidth>
```

Fig. 6. A device configuration expressed in XML

3.3.2 SOAP Message Format

Web services, based on the SOAP message, serve as the communication media. SOAP message is composed of Envelope, Header, and Body elements where Header and Body elements are the sub-elements of Envelope. Header contains the information of message arguments while Body contains remote methods definition and arguments. Figure 7 shows a file sharing SOAP message showing the device owner is Sam Chiang with IP 163.23.199.124 and its file directory information.

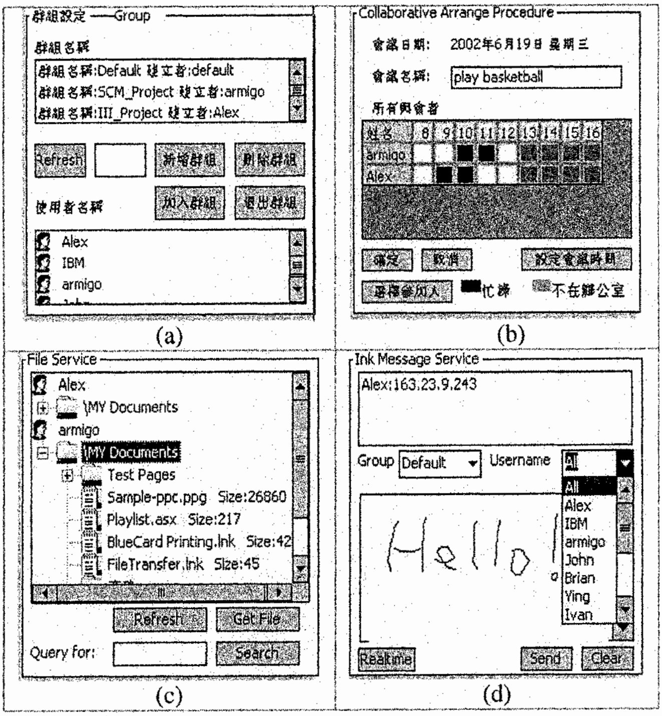
```
<FileSharingInfo xmlns:m='urn:Foo'>
    <Owner ip="163.23.199.124" name="Sam Chiang"/>
        <SharingInfo>
            <DIR name="\MY Documents"/>
            <FILE name="msg.wav" size="2522"/><FILE name="j531.jpg" size="76195"/>
            ...
            <DIR name="\MyMusic">
                <FILE name="Bon Jovi.mp3" size="6671"/>
                ...
            </DIR>
        </SharingInfo>
</FileSharingInfo>
```

Fig. 7. A file sharing SOAP message

4 A Collaborative Mobile Application Scenario

As a concept proof of the proposed Web services based P2P collaborative framework, a collaborative mobile application for smartphones is built on top of the framework. The setup is as followings: one client running on top of iPAQ H3870 was implemented using eVB while another client running on another iPAQ H3870 was implemented using Java (using Jeode JVM). All servers are implemented by Java using Apache Tomcat, Sun Microsystems J2EE, and IBM's UDDI SDK.

Figure 8 shows some interfaces of this collaborative application[§]. Figure 8a offers users and group management with functions of creating, adding, modifying, or deleting users or groups. Figure 8b shows a collaborative scheduling of a group of people. If there is no conflict among all people participated, then a new schedule is created and set. If there is a conflict, events are triggered to notify people with conflicting schedule. Figure 8c shows file sharing service which provides efficient file transfer with auto-reconnection and auto-retransmission. Figure 8d shows ink message service allowing people to communicate with each other using hand written script. Other services include answering machine service, real-time photo taking and transmitting service, Goolge search engine Web service, and so on.



群組名稱:Default 建立者:default
群組名稱:SCM_Project 建立者:armigo
群組名稱:III_Project 建立者:Alex

Alex
IBM
armigo

(a)

會議日期: 2002年6月19日 星期三

會議名稱: play basketball

姓名	8	9	10	11	12	13	14	15	16
armigo									
Alex									

(b)

Alex
\MY Documents
armigo
\MY Documents
Test Pages
Sample-ppc.ppg Size:26860
Playlist.asx Size:217
BlueCard Printing.lnk Size:42
FileTransfer.lnk Size:45

(c)

Alex:163.23.9.243

Group: Default Username: All

All
Alex
IBM
armigo
John
Brian
Ying
Ivan

Hello!

(d)

Fig. 8. Some interfaces of a collaborative application

[§] The snapshots of interfaces are taken from PocketPC simulator for clarity.

5 Conclusion

Because of the rapid development of computer and Internet technologies, the trend towards the integration of wired network, wireless network, and mobile devices is inevitable. Smartphones, with right networking device and access points of Bluetooth, WLAN, and WCDMA networks, are able to connect to the network and interact with other computing devices anytime, anyplace. The authors envision interoperability among smartphones or mobile computing devices will be a critical issue of pervasive computing.

By examining the architectural models of Web services and P2P, and the structural requirements of collaborative mobile applications, this study built a Web services based P2P collaborative framework providing collaborative Web services and P2P facilities to allow smartphones to connect to each other dynamically and to share information seamlessly. As a proof of concept, a collaborative application based on the proposed framework was built where two iPAQ H3870 clients implmented using Java and eVB were ablie to communicate and share information with each other through Web services and P2P. Future work includes incorporating the location information of mobile users to provide a location-aware collaborative P2P framework; providing support for audio/video streaming to enable a realtime video conferencing; adding support for J2ME platform which has been popular lately.

References

1. Pierre, S.: Mobile Computing and Ubiquitous Networking: Concepts, Technologies and Challenges. Telematics and Informatics. 18 (2001) 109-131
2. Kortuem, G.: When Peer-to-Peer Comes Face-to-Face: Collaborative Peer-to-Peer Computing in Mobile Ad-Hoc Networks. Proceedings of the 2001 International Conference on Peer-to-Peer Computing. (2001) 27-29
3. Friday, A., Davies, N., Wallbank, N, Catterall, E., Pink, Stephen: Supporting Service Discovery, Querying and Interaction in Ubiquitous Computing Environments. 10 (2004) 631-641
4. Roman, M.: Gaia: A Middleware Infrastructure to Enable Active Spaces. IEEE Pervasive Computing. 1:4 (2002) 74-83
5. Stoica, I.: Chord: A Scalable Peer-To-Peer Lookup Protocol for Internet Applications. IEEE/ACM Transactions on Networking. 11:1 (2003) 149–160
6. Sutherland, J., Heuvel, J. V. D.: Enterprise Application Integration and Complex Adaptive Systems. Communications of the ACM. 45:10 (2002) 59-64.
7. Mougin, P., Barriolade, C.: Web Services, Business Objects and Component Models. Orchestra Networks, NY (2001)
8. Jopsen, T.: SOAP Cleans Up Interoperability Problems on the Web. IT Professional. 3:1 (2001) 52-55
9. IBM UDDI Service Registry. http://www-3.bim.com/services/uddi/protect/find (2004)
10. Kao, J.: Developer's Guide to Building XML-based Web Services with the Java 2 Platform, Enterprise Edition. Sun Microsystems, CA (2001).

DYNAMIC ROLE BINDING IN A SERVICE ORIENTED ARCHITECTURE

Humberto Nicolás Castejón and Rolv Bræk
NTNU, Department of Telematics, N-7491 Trondheim, Norway
{humberto.castejon, rolv.braek}@item.ntnu.no

Abstract Many services are provided by a structure of service components that are dynamically bound and performed by system components. Service modularity requires that service components can be developed separately, deployed dynamically and then used to provide situated services without undesirable service interactions. In this paper we introduce a two-dimensional approach where service components are roles defined using UML 2.0 collaborations and system components are agents representing domain entities such as users and terminals. The process of dynamic role binding takes place during service execution and provides general mechanisms to handle context dependency, personalisation, resource limitations and compatibility validation. A policy framework for these mechanisms is outlined.

1. Introduction

A *service* may generally be defined as an identified partial functionality provided by a system to an end user, such as a person or other system. The most general form of service involves several system components collaborating on an equal basis to provide the service to one or more users. This understanding of service is quite general and covers both client-server and peer-to-peer services as described in [6]. A common trait of many services is that the structure of collaborating components is dynamic. Links between components are created and deleted dynamically and many services and service features depend on whether the link can be established or not, and define what to do if it cannot be established (e.g. busy treatment in a telephone call). Indeed, setting up links is the goal of some services. For example, the goal of a telephone call is to establish a link between two system components, so that the users they represent can talk to each other. In the past this problem has often been addressed in service specific ways. It may however, be generalised to a problem of *dynamic role binding*, i.e. requesting system components to play roles, such as for example requesting a UserAgent to play the b-subscriber in a call. The response to such a request may be to alert the end-user (if free and available), to reject the call, to forward it or to provide some waiting functionality (if busy). Which feature to select depends on what is subscribed, what other features are active, what resources are available, what the current context is and what the preferences of the user are. By recognising dynamic role binding as a general problem, we believe it is possible to find generic and service independent solutions. In fact, many crucial mechanisms can be associated with dynamic role binding: service discovery; feature negotiation and selection; context dependency resolution; compatibility validation of collaborating service components, and dependency resolution.

Modularity is a well-known approach for easing service development. Service modularity requires a separation of service components from system components, allowing the former ones to be specified and designed separately from the latter ones, then be incrementally deployed and finally be linked dynamically during service execution to provide actual services without undesirable service interactions. We will show that dynamic role binding mechanisms are crucial to achieve the desired separation and modularity and still be able to manage the complex mutual dependencies between service components and system components. It is desirable that such dependencies are not hard coded, but represented by information that can be easily configured and interpreted by general mechanisms, i.e. by some kind of policies.

In this paper we present a service architecture where service modularity and dynamic linking is supported by means of roles and general mechanisms for dynamic role binding. In Sect. 2 the main elements of the architecture are presented: agents mirroring the environment as system components; UML 2.0 collaborations and collaboration roles as service-modelling elements; and UML active classes as service components. In Sect. 3 we show how the proposed architecture provides structure to service-execution policies and how dynamic role binding enables policy-driven feature selection with compatibility guarantees. Finally we conclude with a summary of the presented work.

2. Agent and Role Based Service Architecture

Fig. 1 suggests an architecture for service-oriented systems which is characterized by horizontal and vertical composition. On the horizontal axis, system components are identified that may reside in different computing environments. This axis reflects domain entities (such as users,

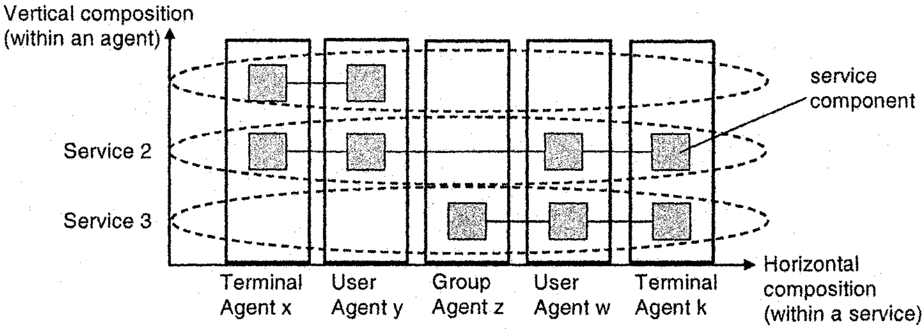


Figure 1. Service Oriented Architecture

user communities and terminals) and resources that must be represented in a service providing system regardless of what services it provides. On the vertical axis, several services and service components are identified that depend on the system components of the architecture. This two-dimensional picture illustrates the crosscutting nature of services which is a well-known challenge in service engineering [4, 8][9, 2].

In the following sections we will present our particular realization of this architecture.

2.1 Agents as System Components

In [6], Bræk and Floch identify two principal system architectures: the *agent oriented* and the *server oriented.* The agent oriented architecture follows the principle that a system should be structured to mirror objects in the domain and environment it serves [3]. This is a general principle known to give stable and adaptable designs. Agents may represent and have clear responsibilities for serving domain/environment entities and resources and thereby provide a single place to resolve dependencies. In the case of personalised communication services accessible over a number of different terminal types, this mirroring leads to a structure of `TerminalAgents` and `UserAgents` as illustrated in Fig. 1. In addition there may be agents corresponding to user communities (e.g. the `GroupAgent` in Fig. 1), service enablers and shared service functionality. Several authors have proposed similar architectures, for example [21] and [1].

Note that such an agent structure reflects properties of the domain being served and not particular implementation details, nor particular services. It is therefore quite stable and service independent. At the

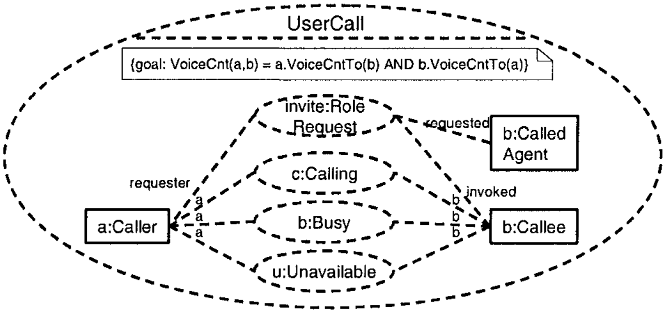


Figure 2. The UserCall Service as a UML 2.0 Collaboration

same time agents provide natural containers for properties and policies of domain entities like users, terminals and service enablers.

2.2 UML Collaborations as Services and Roles as Service Components

As illustrated in Fig. 1, a service is a partial functionality provided by a collaboration between service components executed by agents to achieve a desired goal for the end users. UML 2.0 *collaborations* [13] are well suited for service modelling as they are intended to describe partial functionalities provided by collaborating roles played by objects. An interesting characteristic of UML collaborations is that they can be applied as *collaboration uses* and employed as components in the definition of larger collaborations. This feature enables a compositional and incremental design of services, as we explain in [17, 7]. For instance, Fig. 2 shows a collaboration specifying a `UserCall` service in terms of collaboration roles linked by collaboration uses, which correspond to different phases and features of the service. It also shows a simple goal expression representing a desired goal state for the collaboration. Behaviour descriptions can be associated with the collaboration to precisely define the service behaviour, including precise definitions of the visible interface behaviour that objects must show in order to participate in the collaboration. Collaborations thereby provide a mechanism to define *semantic interfaces* that can be used for service discovery and to ensure compatibility with respect to safety and liveness (i.e. reaching the desired goal states) when linking service components, as we discuss in [18].

Ideally, service models should be independent of particular system structures. It can be argued, however, that it is necessary and beneficial to take a minimum of architectural aspects into account [20]. The challenge is to do this at an abstraction level that fits the nature of the services without unduly binding design solutions and implementations. In our architecture (see Fig. 1) the horizontal axis represents the agent structure and the vertical axis represents the services modelled as collaborations with roles that are bound to agents. The service-independent agent structure is therefore instrumental, since it helps to identify and shape roles, without introducing undue bindings to implementation details. At the same time it provides an architectural framework for role composition, role binding and role execution.

In this service oriented architecture, service specific behaviour is the responsibility of (service-) roles while domain specific behaviour and policies are the responsibility of agents. Interactions between roles and agents are needed primarily in the process of creating and releasing dynamic links, that is, the process of *dynamic role binding.*

Roles need to be mapped to well defined service components, which can then be deployed and composed in agents to provide (new) services without causing safety or liveness problems. We do this by defining service components as UML active classes with behaviour defined by state machines. Note that service components may implement one or more UML collaboration roles composed by means of collaboration uses, as it is roughly illustrated in Figs. 2 and 4. Each of these collaboration roles will correspond to a different service or service feature. We assume that service components are typed with semantic interfaces with well defined feature sets. This information is exploited when service components are dynamically linked within a service collaboration in order to ensure their compatibility in terms of safety and liveness criteria, as explained in [18]. In addition, it is necessary to ensure that the service components can actually be bound to the intended agents. Their feature sets may also be restricted and dynamically selected during the binding process. These aspects will be discussed in the following sections.

Note that, for the sake of simplicity, in the rest of the paper we will use the word *role* to name service components.

2.3 Dynamic Role Binding

Dynamic Role binding has three distinct phases:

1 *Agent identification,* which aims at identifying an agent by consulting a nameserver or performing a service discovery. Some service features are related to the agent identification, e.g. aliasing, business domain restrictions or originating and terminating screening features in telephony.

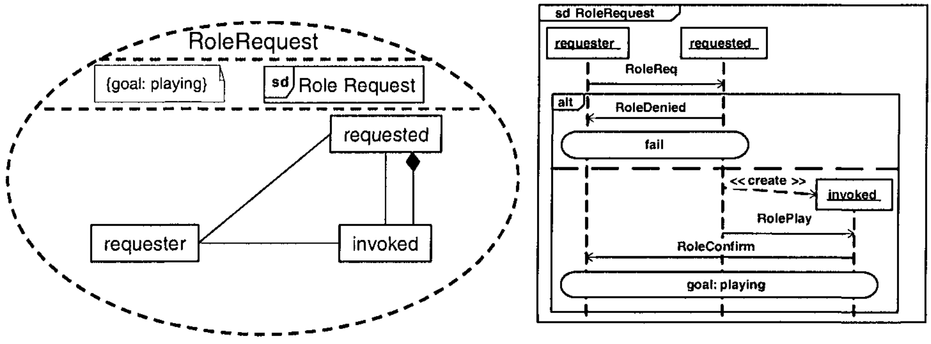


Figure 3. Role Request Pattern

2 *Role request,* which aims at creating a dynamic link according to a semantic interface with an agreed feature set. This means to request the agent identified in phase 1 to play a role with a certain feature set, which can be negotiated. The role with the agreed feature set is finally invoked. The role request pattern [5] described in Fig. 3 provides one partial solution to this. Using this protocol any role can request an agent to play a complimentary role. If the agent is able to play the requested role, it invokes it and a link is dynamically established between the requesting and the requested roles, so that they can collaborate. This is illustrated in Fig. 2, where a `UserAgent` is requested (invited) to play the `b` role in a `UserCall` collaboration. The response of the `UserAgent` in this case is to enable one of three features, represented as collaboration uses: `Calling`, `Busy` or `Unavailable`.

3 *Role release,* which signals that a role is finished and has released whatever resources it had occupied.

Once a role is invoked it can proceed autonomously, until it reaches a state where interaction with agents is again required for one of the following reasons:

- it needs to bring a new role into the collaboration (i.e. create another dynamic link). In this case it first needs to identify the agent that should play the role and then initiate a role request to this agent, as explained above;
- it needs to check what feature or feature set to select at a certain point in its behaviour, if this depend on agent policy (e.g. if mid-call telephony services are allowed);

- it needs to signal to its own agent that it is available for additional linking, in response to an incoming role request (e.g. to perform a call waiting feature); or

- it is finished and performs role release.

Note that all cases except the second are related to dynamic role binding. Also note that a large proportion of features are discovered, selected and initiated in connection with dynamic role binding.

3. Governing Service Execution with Policies

Dynamic role binding is clearly a very central mechanism in service execution. As we have already pointed out, associated with dynamic role binding are such key issues as: service discovery, feature negotiation and selection, context dependency resolution, compatibility validation and feature interaction detection/avoidance. The challenge is to find general, scalable and adaptable ways of dealing with those issues.

In general a role can only be bound to an agent without undesired side-effects if certain (pre-/post-) conditions hold. By explicitly expressing these conditions as constraints, we may check them upon role-binding and only allow the role to be invoked if they are satisfied. It is also important to give users the possibility to express their preferences to control the selection of features when, for example, the requested service can not be delivered. In the following we will use the term *policy* to cover both general role binding constraints and user preferences. In doing that we adhere to the usual definition of policy that can be found in the computer-science literature: *a rule or information that modifies or defines a choice in the behavior of a system* [11].

The agent architecture discussed in the previous section provides a natural way to structure policies into three groups:

- *Role-binding* policies, which constrain the binding of roles to agents at run-time.

- *Collaboration* policies, which express constraints that must hold for a collaboration (i.e. a service) as a whole when it is executed. They aim at preventing actions that may compromise the intentions and goals of the collaboration.

- *Feature-selection* policies, which control the triggering of context-dependent service features.

We will take a closer look at each of these policies in the following.

3.1 Role-binding and Collaboration Policies

Role-binding policies represent conditions that must be satisfied for a role to be bound to an agent. These policies may be associated with agent types, so they shall hold for all instances of that type, or they may be defined for specific agent instances (usually describing user preferences and/or user permissions). Finally, role-binding policies may also be associated with role types and they shall hold for all instances of that role type.

The role-binding policies associated with a role type define constraints that the role imposes on any agent it may be bound to and, thereby, indirectly on system resources. For example, the role-binding policy of a role may require that role to be bound to a `TerminalAgent` representing a specific terminal type with specific capabilities (e.g. a PDA).

The role-binding policies associated with an agent represent, on the contrary, constraints that the agent imposes on the roles it can play. When these policies are associated with an agent type, they represent constraints on the type and multiplicity of the roles that can be bound to the agent, as well as other constraints imposed by the service provider (e.g. that the user must hold a valid subscription to play a certain role). When they are associated with a particular agent instance, they represent user preferences and/or user permissions specifying when that particular agent should or should not play a certain role. These preferences/permissions can be seen to express context dependency (e.g. on location, calendar, presence or availability). For example, a user may define a role-binding policy for her `UserAgent` to express that it should only participate in a `UserCall` service, playing the `callee` role, if the invitation was received between 8 am and 11 pm.

Collaboration policies express constraints that must hold for a collaboration (i.e. a service) as a whole when it occurs. We may associate collaboration policies with a UML collaboration, so that they shall hold for all occurrences of that collaboration. For instance, a collaboration policy may be associated with a conference-call collaboration to prohibit the agent playing the conference-controller role to temporally interrupt its participation in the service. This policy would specifically prohibit the conference-controller role to invoke the *hold* feature. Agent instances may also hold specialised collaboration policies that, in this case, shall only be satisfied for those occurrences of the collaboration where the agent participates. These policies may then represent user preferences. An important use of such user-defined collaboration policies is to constrain who participates in a service session. For personal communication services the identity of the agents participating in a service is important (e.g. a calling user wants a specific user's UserAgent to play the b role - see Fig. 2). It is not only important who must be invited to a service, but also who cannot be invited. Some users may not want to talk to certain people, or they may not like, for example, to talk to a machine. We can easily solve this problem using collaboration policies by constraining the type and/or id of agents that can participate in a service session, as well as the roles they can play. For instance, if a user does not want to be redirected to an automatic-response machine, she may define a collaboration policy for the UserCall service constraining the participation of IVRAgents[1]. This policy would be held by her UserAgent, thus only affecting UserCall services in which she may participate.

Role-binding and collaboration policies are checked upon a role request by both the requester and the requested agents. The requester agent checks the collaboration policies associated to the service being (or to be) executed before the role request is sent. This is done to confirm that inviting the requested agent would not violate those policies (e.g. that a UserAgent representing an undesired user would not be invited when performing a forwarding). At the reception of the role request, the requested agent checks first the collaboration policies for conformance on joining the collaboration (e.g. to ensure that all other participating agents are welcome). Thereafter, it checks the role-binding policies concerning the requested role, which is only bound if those policies are satisfied.

Policies defined for a collaboration (and its roles) are "inherited" when that collaboration is employed, as a collaboration use, in the specification of other collaborations. Therefore, when a collaboration is bound to a set of agents for its execution, all policies defined for the collaboration itself and for its sub-collaborations must hold. This is illustrated in Fig. 4. The upper part shows a collaboration specifying a FullCall service. This service is a collaboration between four roles and it is composed from the UserCall service described in Fig. 2 and two uses of a TermCall service, which specifies the collaboration between UserAgents and TerminalAgents. Policies have been defined for each of the roles and collaboration uses in FullCall (P2-P8). In addition a policy (P1) has been defined for the FullCall collaboration as a whole. The lower part of the figure illustrates a set of UserAgents and TerminalAgents, representing users and terminals, performing the FullCall service. It is important to note that for this collaboration use (i.e. fcx:FullCall) all and each of the policies defined for the FullCall collaboration must hold. This is indicated by the annotation P1+P2+...+P8. Moreover, each agent holds a set of role-binding, collaboration and feature selection policies, called P10-P13 in the figure, that also must hold in the execution

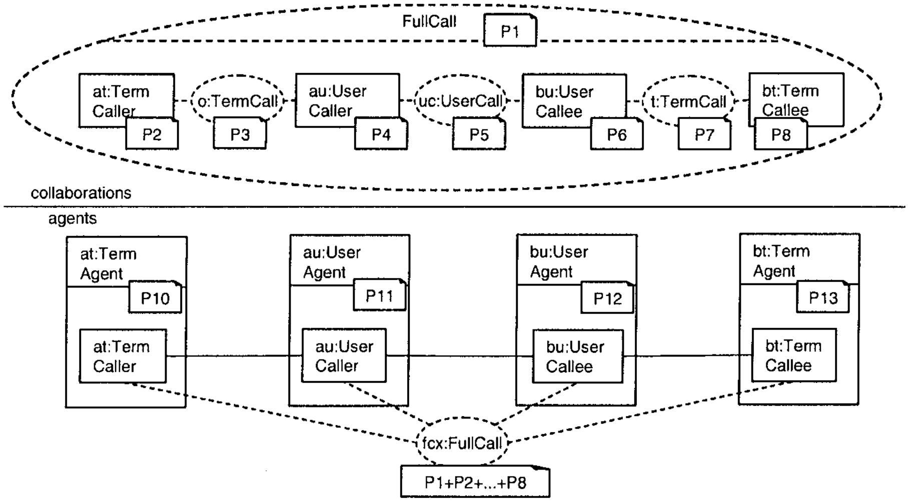


Figure 4. Role Binding

of `fcx:FullCall`. Therefore, if for example `at:TermCaller` had some collaboration policy that would not hold when inviting `bt:TermCallee`, `fcx:FullCall` could not be completed.

3.2 Feature Selection Policies

A feature can be defined as a unit of functionality in a base service. In general, we can differentiate two types of features, depending on how they are selected and triggered:

- features that are triggered within a role as part of its behaviour (e.g. call-transfer)
- features that are triggered upon role-binding depending on the agent's context and policy (e.g. call-forward on busy subscriber)

We refer to the first type of features as *mid-role-triggered* (or just *mid-role*), and to the second as *role-binding-triggered.* Mid-role-triggered features are selected as part of the role behaviour. If they may be disabled by policy decisions this should be agreed during a negotiation phase before the role is bound. Alternatively, the role may consult its containing agent concerning actual policies before invoking mid-role features. One example of such a feature is call-forward on no-answer. It describes an alternative behavior to the `UserCall` service and it is triggered when the latter does not achieve its goal (i.e. contacting the end-user).

Role-binding-triggered feature selection occurs when an agent receives a role request. In that case the agent (1) checks its feature selection policies to determine if, in the current context, there is an alternative feature to be selected, without even trying to invoke the requested role. This may be the case if the following feature selection policy existed: "when the `b` role is requested in a `UserCall` and the (called) user is not at home, always select call-forward instead". This checking returns either the requested role (if no feature selection policy was satisfied) or an alternative one. The selected role is then target (2) for checking the collaboration and role binding policies to decide whether it may be actually invoked. If yes, a confirmation message is sent back to the requesting role. Note that if the role that is finally selected to be invoked is not the originally requested one, the confirmation message may be replaced by a negotiation phase (not shown in Fig. 3). If otherwise collaboration and/or role binding policies are not satisfied, (3) a search is again performed for a substitute role that may be invoked, and, if found, the process is repeated from (2), until a role with specific features is agreed and invoked. In addition, if an invoked role does not achieve its goal during the service execution, a search for an alternative role, implementing a mid-role-triggered feature, can be made once more (e.g. to invoke call-forward on no-answer).

From the above explanation three generic events can be distinguished that trigger the selection of features describing alternative behavior. These events are:

- *OnRoleRequest,*
- *OnUnsuccessfulRoleBinding*, and
- *OnNonAchievedGoal* event.

Feature selection policies can then be defined, by for example end-users, as event-condition-action (ECA) rules, where the event is one of the three just mentioned, the condition is expressed in terms of the context and the action is the selection of a feature.

Note that up to now we have just talked about the use of feature selection policies to select features of a base service. However their potential is actually greater than that. There is nothing that prevents us from using feature selection policies to specify any service as an alternative to another one. That is, we may specify which event and condition leads to the substitution of a role X for a role Y, where roles X and Y are not necessarily related. In this case, the role at the requesting side must most likely be also substituted. A negotiation between the parties would then be necessary.

The use of policies for service-execution management and personalization is not novel. For example, the Call Processing Language (CPL) [10] is used to describe and control Internet telephony services. With CPL users can themselves specify their preferences for service execution. Reiff-Marganiec and Turner [16] also propose the use of policies to enhance and control call-related features. The novelty of our work lies in the structuring of policies we make, based on the proposed service architecture.

4. Conclusion

We have presented a two-dimensional service oriented architecture where service components are roles defined using UML 2.0 collaborations and system components are agents representing domain entities such as users and terminals. Service modularity is achieved by the separation of service components from system components, and by general policy-driven mechanisms for dynamic role binding that handle context dependency, personalisation, resource limitations and compatibility validation. Central parts of this architecture, such as the role request pattern and a simple form of XML-based role-binding policy, have been implemented in ServiceFrame [5] and have been used to develop numerous demonstrator services within the Program for Advanced Telecommunication Services (PATS) research program [14], which is a cooperation between the Norwegian University of Science and Technology (NTNU), Ericsson, Telenor and Compaq (now Hewlett-Packard). These experiments have confirmed that dynamic role binding is central not only to traditional telecom services, but also to a wide range of convergent services, and that explicit support for role-binding helps to manage the complexity of such services. The use of more advanced role-binding policies specified as BeanShell [12] scripts has also been studied in [19]. At the time of writing this paper, ServiceFrame has been extended with support for java-based role-binding, collaboration and feature selection policies that can be specified by both end-users and service providers to handle context dependency [15].

An interesting problem that has not been treated is undesirable interactions between two or more roles simultaneously played by an agent in different services. This is known as the feature interaction problem. We believe that our policy-driven mechanisms for dynamic role binding can help to avoid such interactions, if the agent maintains the consistency between the policies imposed in different services. We are also investigating in this direction.

Acknowledgments

We wish to thank NTNU and the ARTS project for partial funding this work. G. Melby and K.E. Husa at Ericsson have been extremely valuable through numerous discussions and by implementing the core of the architecture. We are also grateful to the many MSc students that have contributed by making services using these principles and by proposing and prototyping role binding mechanisms. Finally we appreciate all the input from our close colleagues J. Floch, R. T. Sanders, F. Krämer and H. Samset.

Notes

1. IVR stands for Interactive Voice Response machine

References

[1] Amer, M., Karmouch, A., Gray, T. and Mankovski, S. (2000). Feature-interaction resolution using fuzzy policies. In *Feature Interactions in Telecommunications and Software Systems VI*, pages 94–112, Glasgow, Scotland, UK.

[2] Bordeleau, F., Corriveau, J. P. and Selic, B. (2000). A scenario-based approach to hierarchical state machine design. In *ISORC '00: Proceedings of the Third IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, page 78. IEEE Computer Society.

[3] Bræk, R. and Haugen, Ø. (1993). *Engineering Real Time Systems. An object-oriented methodology using SDL.* Prentice Hall.

[4] Bræk, R. (1999). Using roles with types and objects for service development. In *IFIP TC6 WG6.7 Fifth International Conference on Intelligence in Networks (SMARTNET)*, pages 265–278, Pathumthani, Thailand. Kluwer.

[5] Bræk, R., Husa, K. E. and Melby, G. (2002). ServiceFrame: WhitePaper. *White paper*, Ericsson Norarc. Available at: http://www.pats.no/devzone/platforms/ServiceFrame/doc/ServiceFrameWhitepaperv8.pdf.

[6] Bræk, R. and Floch, J. (2004). ICT convergence: Modeling issues. In *System Analysis and Modeling (SAM), 4th International SDL and MSC Workshop*, pages 237–256, Ottawa, Canada.

[7] Castejón, H. N. (2005). Synthesizing state-machine behaviour from UML collaborations and Use Case Maps. In Prinz, Andreas, Reed, Rick, and Reed, Jeanne, editors, *12th SDL Forum*, volume 3530 of *Lecture Notes in Computer Science*, pages 339–359, Grimstad, Norway. Springer.

[8] Floch, J. (2003). *Towards Plug-and-Play Services: Design and Validation using Roles.* PhD thesis, Department of Telematics, Norwegain Univ. Science and Technology, Trondheim, Norway.

[9] Krüger, I. H., Gupta, D., Mathew, R., Moorthy, P., Phillips, W., Rittmann, S. and Ahluwalia, J. (2004). Towards a process and tool-chain for service-oriented automotive software engineering. In *Proceedings of the ICSE 2004 Workshop on Software Engineering for Automotive Systems (SEAS).*

[10] Lennox, J., Wu, X. and Schulzrinne, H. (2004). Call Processing Language (CPL): A language for user control of internet telephony services. RFC 3880, IETF.

[11] Lupu, E. C. and Sloman, M. (1999). Conflicts in policy-based distributed systems management. *IEEE Trans. Softw. Eng.*, 25(6):852–869.

[12] Niemeyer, P. (1997). BeanShell - lightweight scripting for java. Available at: http://www.beanshell.org/.

[13] Object Management Group (2004). *UML 2.0 Superstructure Specification.*

[14] Program for Advanced Telecom Services (PATS). Accessible at: http://www.pats.no.

[15] Pham, Q. T. (2005). Policy-based service personalization. Master's thesis, Dept. of Telematics, Norwegian University of Science and Technology (NTNU).

[16] Reiff-Marganiec, S. and Turner, K. J. (2003). A policy architecture for enhancing and controlling features. In *Feature Interactions in Telecommunications and Software Systems VII*, pages 239–246, Ottawa, Canada.

[17] Sanders, R. T., Castejón, H. N., Kraemer, F. A. and Bræk, R. (2005). Using UML 2.0 collaborations for compositional service specification. In *ACM/IEEE 8th International Conference on Model Driven Engineering Languages and Systems (MoDELS)*, Montego Bay, Jamaica.

[18] Sanders, R. T., Bræk, R., Bochmann, G. v. and Amyot, D. (2005). Service discovery and component reuse with semantic interfaces. In *12th SDL Forum*, Grimstad, Norway.

[19] Støyle, A. K. (2003). Flexible user agent. Technical report, Dept. of Telematics, Norwegian University of Science and Technology (NTNU).

[20] Zave, P. (2003). Feature disambiguation. In *Feature Interactions in Telecommunications and Software Systems VII*, pages 3–9, Ottawa, Canada.

[21] Zibman, I., Woolf, C., O'Reilly, P., Strickland, L., Willis, D. and Visser, J. (1995). Minimizing feature interactions: An architecture and processing model approach. In *Feature Interactions in Telecommunications III*, pages 65–83, Kyoto, Japan.

FORMAL MODELLING OF AN ADAPTABLE SERVICE SYSTEM

Mazen Malek Shiaa, Finn Arve Aagesen, and Cyril Carrez
NTNU, Department of Telematics, N-7491 Trondheim, Norway
{malek, finnarve, carrez}@item.ntnu.no

Abstract: Adaptable service systems are service systems that adapt dynamically to changes in both time and position related to users, nodes, capabilities, status, and changed service requirements. We present a formal model for the basic entity used for the implementation of the service functionality in the Telematics Architecture for Play-based Adaptable Service systems (TAPAS). This basic entity is the *role-figure*, which executes in the nodes of the network. The formal model is denoted as the *role-figure model.* It comprises behaviour specification, interfaces, capabilities, queue of messages, and executing methods for role-figures. Its semantics is based on an ODP (Open Distributed Processing) semantic model and rewriting logic, and is used to prove properties such as: *plug ability*, *consumption ability*, and *play ability.*

1. INTRODUCTION

Service systems consisting of services realized by service components are considered. Service components are executed as software components in network nodes and terminals. A terminal is a node operated by a human user. Those generic components are denoted as *actors*. This name comes from the analogy with the actor in the theatre, where an *actor* plays a *role* in a *play* defined by a *manuscript.* We use *role-figure* as a generic concept for the *actor* which is playing a *role.* So services and service components are constituted by *role-figures*. The attributes of services, service components and nodes are formalised, stored and made available using the concepts of *status* and *capability. Status* is a measure for the situation in a system concerning the number of active entities, traffic, and Quality of Service (QoS). *Capability* is the properties of a node or a user defining the ability to do something.

Telematics Architecture for Play-based Adaptable Service systems (TAPAS) is a research project which aims at developing an *architecture* for adaptable service systems. *Adaptable* means that the service systems will adapt dynamically to changes in both time and position related to users, nodes, capabilities, status, and changed service requirements. In TAPAS, adaptability is modelled as a 3-classes property: *1)* Rearrangement flexibility, *2)* Failure robustness and survivability, and *3)* QoS awareness and resource control [1,2,3]. One objective is to gain experiences by implementing those various features. Parts of the specified functionality have been implemented based on java and web services platforms. The TAPAS architecture has been specified using various UML diagrams.

However, it has been realized that the behaviour parts of the architecture lacks a formal foundation. The implementation software only contains program code, while the UML diagrams only specify parts of the functionality informally. We need a model that can be used as a basis for the formal verification of the various issues related to adaptability. Mainly, this means that when a service is trying to adapt to a change in the service system, it will change some of its composing parts (for example by moving or creating new service components). We would like the formal model to ensure that the actions taken by the service will achieve its goal, and without harming the whole architecture. In this paper we present a formal model of the main component of the TAPAS architecture. This component is the role-figure and the formal model is denoted as the *role-figure model*. The model will be used to verify the behaviour of the role-figures, and will be the basis for the formal verification of certain properties of the system.

Related works and TAPAS are discussed in Sec. 2 and 3, respectively. The semantics of the role-figure model is presented in Sec. 4, while its properties are discussed in Sec. 5. Section 6 concludes the paper.

2. RELATED WORK

The role-figure model must capture the features and properties of service adaptability. Various formal frameworks have been considered as candidates. Process Algebras such as π-calculus [4] have very powerful notations which abstract system elements in terms of processes and communication channels, focusing on the sequence of inputs/outputs. Specifying the TAPAS architecture and role-figures using process algebras is possible, but the specification would be very detailed and lengthy. Moreover, we are more concerned with the constructive states of the system than the input/output sequences.

The ODP framework and the ODP formal model presented by Najm and Stefani in [5], and further elaborated with Dustzadeh [6] is also very interesting. ODP computational objects have states and can interact with their environment through operations on interfaces. The object interfaces and operations provide an abstract view of the state of the object. Access to the object is only possible through invocations of its advertised operations on a designated interface. ODP computational objects and role-figures have several similarities, e.g. the definition of interfaces and their dynamic creation, as well as the distributed operation invocation.

The ODP formal model was based on rewriting logic theory [7]. The semantics of the Rewriting Logic is based on the models of rewrite systems: it is applied to terms which are rewritten based on rewriting rules of the form $t \rightarrow t'$ (meaning the term *t* is rewritten to *t'*). This theory has been used for the formal specification and verification of many other systems, such as the Actor semantics [8], and the formalization of active network [9,10].

Our role-figure model is based on the ODP formal semantics, and Rewriting Logic.

3. TAPAS ARCHITECTURE

In accordance with TINA architectural framework, TAPAS is separated into two parts: the *computing* architecture and the *system management* architecture. The *computing* architecture is a generic architecture for the modelling of any service system. The *system management* architecture, not detailed in this paper, is the structure of services and management components.

The computing architecture has three views: the *service* view, the *play* view, and the *network* view (*Figure 1*). The *service* view concepts are generic and should be consistent with any service architecture. Basically, a service system consists of several service components.

The *play* view concepts are the basis for implementing the service view concepts. The concepts of actor, role, role figure, manuscript, capability and status have already been defined. Additional concepts are *director*, *role-session* and *domains*. The *director* acts according to a special role and manages the performance of different role-figures involved in a certain *play*. It also represents a *play domain*. *Role-session* is the projection of the behaviour of a role-figure with respect to one of its interacting role-figures.

The *network* view concepts are the basis for implementing the play view concepts. In the network view, *capability* is provided by a *node* or is owned by a *user*. *User*, *node*, and *capability* have *status* information. A *play domain* may be related to one or more *network domain* (a set of *nodes*), as a play may be distributed over several network domains.

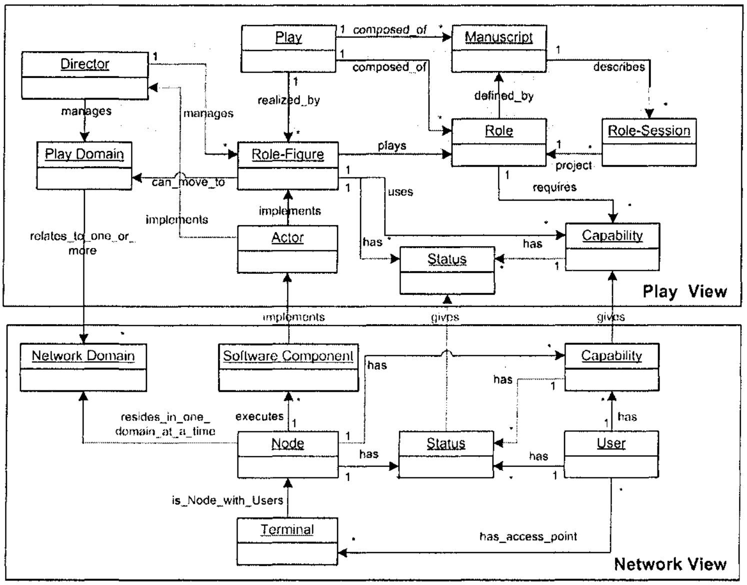


Figure 1. Computing architecture – Play View and Network View

The play view intends to be a basis for designing functionality that can meet the requirements related to rearrangement flexibility, failure robustness and survivability, and QoS awareness and resource control. The play view concepts allow service components to be instantiated according to the available capabilities and the status in the network. They also facilitate the handling of dynamic changes in the installed service components, which occur due to changing capabilities, changing functionality, changing locations, etc.

An important concept related to the *role-session* is the *interface.* Two role-figures can only communicate if they are connected via interfaces. A role-figure creates an interface locally and connects it to another interface in another role-figure. Sending a message is performed by the local interfaces of a role-figure. TAPAS core platform is a platform supporting the functionality of the play view by offering a set of methods. Role-figures will also interact with each other via *signals* that are used to interact with the behaviour of the role-figure, and thus performing the service.

The role-figure model is a formal model of the role-figure implementation. The following aspects need to be included:

- Role-figures are realized by actors and can be dynamically created;
- Role-figures interact via role-sessions and are connected via interfaces. Messages are asynchronous;

- Role-figures comprise behaviour (an extended finite state machine), and methods used for management and control of actor objects;
- Messages are: signals (used to interact with the role-figure behaviour), requests to invoke methods, and returns (results of the invoked methods);
- The main role-figure methods are:

– *PlugInActor*	instantiates role-figures
– *PlugOutActor*	terminates role-figures
– *CreateInterface*	creates interfaces in role-figures
– *BehaviourChange*	changes role-figure behaviour
– *CapabilityChange*	adds or modifies capabilities
– *RoleFigureMove*	moves role-figure to new locations.

The *RoleFigureMove* procedure is used to implement the role-figure mobility. We believe this mobility is one of the keys to adaptability. To ensure that the moving role-figure will continue its execution after the movement, the following parts of the role-figure will be moved as well:

- *behaviour* described by a specification
- *role-sessions* and *interfaces* with other role-figures
- consumed *capabilities* in the node
- *queue* of incoming messages
- executing *methods* (or the role-figure active tasks)

In this paper we only handle the first three parts: behaviour, interfaces, and capabilities. Role-figure mobility management is further detailed in [11].

4. THE ROLE-FIGURE MODEL

This section presents the semantics of the role-figure model. These are semantic rules defining the structure and the behaviour of role-figures. These rules are inspired by the semantics of the ODP computational model [5], based on the rewriting logic [7]. We will use the notations:

a, ***b***, ***f***, ***g***, ***h***	role-figure names;
A,A′,… B,B′,…	role-figures ***a*** and ***b*** as they evolve, respectively;
i: α, j: α'	interface names i and j, with their types α and α';
$r=\langle w_1=v_1,w_2=v_2\rangle$	record with fields w_1 and w_2, having values v_1 and v_2; $r.w_1$ will be used to access the value of w_1 in r;
$\|\|$	asynchronous parallel operator;
$\lhd$	insert operator; "a$\lhd$b" only executes if a is not in b.
$\rhd$	remove operator. "a$\rhd$b" only executes if a is in b.

The operators $||$, $\lhd$ and $\rhd$ are associative and commutative, with $\varnothing$ as neutral element.

4.1 Role-figure components

The semantics for the role-figure model is based on a *Role-Figure Configuration (RFC)*, which is a set of role-figures interacting asynchronously:

$$
\begin{array}{lll}
\mathrm{RFC} & ::= & \varnothing \mid \mathrm{RFCE} \mid \mathrm{RFC} \parallel \mathrm{RFC} \\
\mathrm{RFCE} & ::= & \mathrm{RF} \mid \mathrm{MSG} \\
\mathrm{RF} & ::= & \langle \mathrm{Name} = \mathrm{string}, \mathrm{Int} = \gamma, \mathrm{Beh} = \beta, \mathrm{Cap} = \pi \rangle \\
\mathrm{MSG} & ::= & \mathrm{Req} \mid \mathrm{Sig} \mid \mathrm{Ret}
\end{array}
$$

A role-figure configuration *RFC* is composed of parallel RFC Elements (*RFCE)*, which is either a role-figure *RF*, or a message *MSG.* Three kinds of message exist: a method invocation request *Req*, a communicating signal *Sig*, and a method return result *Ret.* A role-figure *RF* has a name (*Name*) and is defined by a set of interfaces (*Int*), a behaviour (*Beh*), and a set of capabilities (*Cap*). These parts (except *Name*) may evolve as the role-figure consumes messages. Role-figure names are used to distinguish different role-figures; however, as a simplification, we will omit this name in the rest of the article and assume $A, A', \ldots$ always stand for role-figure $\boldsymbol{a}$. The definitions of the role-figure are the following:

Interface	γ	$::= \varnothing \mid \gamma \lhd [j{:}\alpha] \mid \gamma \rhd [j{:}\alpha]$
Behaviour	β	
Capabilities	π	$::= \varnothing \mid \pi \lhd [c{:}cn] \mid \pi \rhd [c{:}cn]$
Invocation req.	Req	$::= \langle tar = j{:}\alpha,\ src = a,\ met = m{:}mn,\ ret = r,\ arg = p \rangle$
Signal	Sig	$::= \langle tar = j{:}\alpha,\ src = a,\ name = sig,\ arg = p \rangle$
Return	Ret	$::= \langle tar = j{:}\alpha,\ src = a,\ arg = p \rangle$
Argument list	p	$::= (p_1{:}t_{1,}, \ldots, p_n{:}t_n)$

Where:

γ list of interface definition $[j{:}\alpha]$ where j is an interface reference of type α. Types of interfaces are discussed in the next paragraph;

β behaviour, based on an EFSM specification (see next paragraph);

π list of capabilities $[c{:}cn]$. cn is a name denoting the type of capability. c denotes the capability identifier which is an instance or value of cn.

Req method invocation request sent by the role-figure *src* to the target interface *tar,* invoking method *met* with arguments *arg* and return *ret.*

Sig signal called *name*, sent by the role-figure *src* to the target interface *tar* with argument list *arg.*

Ret return from a method invocation sent by the source role-figure *src* to the target interface *tar*, with argument list *arg.*

p argument list of parameters $p_1, \ldots, p_n$ with types $t_1, \ldots, t_n$, respectively.

Interface types are defined as follow:

$\alpha \quad ::= \langle m_1{:}\ methsig, \ldots, m_n{:}\ methsig, sig_1, \ldots, sig_k \rangle$

$methsig ::= Nil \mid p \rightarrow return$ with p an argument list as defined earlier

Where:

$m_1, .., m_n$ Method names;
methsig Method signature with arguments *argument* and a return *return*;
$Sig_1,..,sig_k$ signals with a name and arguments (i.e. like *Sig*, without *src, tar*)

The behaviour definition, *β*, is based on the operational semantics of the state machine model. We added to this EFSM the semantic the notion of stable states, which are states where a behaviour change is allowed:

$$\beta ::= \langle B = b : behaviour,\ St = st : state,\ Sg : g,\ Sc : s,\ Ss : s\rangle$$

$s ::= (state_1, ..., state_f)$ state names
$g ::= (sig, ..., sig_f)$ signal names

Where:

B EFSM behaviour specification containing the state transition rules: triggering events, tasks performed, signals sent, and next states
St current state
Sg set of input signals (trigger events for state transition at current state)
Sc set of successor states (next states after the firing of input signals)
Ss Set of stable states (states where behaviour change is permitted)

As a role-figure behaviour evolves and transits from one state to another, *St, Sg, Sc,* and *Ss* change and reflect the status of the role-figure behaviour.

4.2 Behaviour evolution

This section describes the set of rewriting rules that handle the behaviour of a role-figure. The general form of the rewriting rules for role-figure ***a*** is the following[1]:

$$1:\ A \,||\, T \,||\, \Theta \,||\, M \longrightarrow A' \,||\, \Sigma \,||\, T' \,||\, \Theta' \,||\, M' \quad \text{if } C$$

Where:

l is a label. *A* and *A'* stand for role-figure ***a*** that evolves from *A* to *A'*. *Σ* is the role-figures created in this rewriting rule (e.g. *Σ* can be *B* meaning that role-figure ***b*** was created). *T* and *T'* are *return* sets, *Θ* and *Θ'* are *signal* sets, *M* and *M'* are *request* sets. *C* is a condition.

This general rewriting rule, inspired from [5], is used to handle the transitions of any role-figure configuration. As such, a number of role-figures and messages (signals, requests and returns) can come together and participate in a transition in which some new role-figures and new messages may be created. Some restrictions apply:

- messages (*returns, signals*, and *requests*) are all consumed in a transition: $$T \cap T' = \Theta \cap \Theta' = M \cap M' = \emptyset$$
- created role-figures are unique: $B \in \Sigma$ implies ***b*** is unique

[1] Recall that *A* and *B* define the role-figure elements RF whose names are ***a*** and ***b***.

- created messages have their *src* field set to the role-figure that sent them, and *tar* is connected to an interface of an existing role-figure:

  If $msg \in M' \cup T' \cup \Theta'$ with $msg.tar = [i{:}\alpha]$, then:
  $msg.src = \boldsymbol{a} \;\wedge\; [i{:}\alpha] \in A.Int$
  $\wedge\; \exists\, \boldsymbol{b} \in RoleFigures$[2], $j{:}\alpha \in B.Int$ such that $\mathtt{connected}(i,j)=\mathrm{TRUE}$
- Messages must be received by the proper interface indicated in *tar*:

  If $msg \in M \cup T \cup \Theta$ with $msg.tar = [i{:}\alpha]$, then:
  $\exists\, [j{:}\alpha] \in A.Int$ with $\mathtt{connected}(i,j)=\mathrm{TRUE}$

The predicate $\mathtt{connected}(i,j)$ checks that interfaces i, j are interconnected. This issue is left opened so no restriction is made on future implementations (for example, i can be made of the addresses of the local interface and the distant one j).

The rewriting rules will handle behaviour evolution, communications and adaptability functionality. From now on, the role-figure parts will remain constant when applying the rules unless mentioned otherwise.

The following set of rules handle behaviour evolution (internal_action) and communication between role-figures:

internal_action: $A \rightarrow A'$
with: $A.Cap \subseteq A'.Cap \;\wedge\; A'.Beh.St \in A.Beh.Sc \cup \{A.Beh.St\}$

send_request: $A \rightarrow A' \,\|\, req$
Assume $req = \langle tar = i:\alpha, src = a, met = m:mn, ret = r, arg = \tilde{p} \rangle$:
$mn\colon args_m \rightarrow return_m \in \alpha$ and $r = return_m \wedge \tilde{p} = args_m$

recv_request: $A \,\|\, req \rightarrow A'$
Assume $ret = \langle tar = i:\alpha, src = a, ret = r, arg = \tilde{p} \rangle$:
$mn\colon args_m \rightarrow return_m \in \alpha$ and $r = return_m \wedge \tilde{p} = args_m$

send_return: $A \rightarrow A' \,\|\, ret$
recv_return: $A \,\|\, ret \rightarrow A'$
send_signal: $A \rightarrow A' \,\|\, sig$
Assume $sig = \langle tar = i:\alpha, src = a, name = sig, arg = \tilde{p} \rangle$:
$\exists\, sig_k \in \alpha$ such that $sig_k.name = sig$

recv_signal: $A \,\|\, sig \rightarrow A'$
$sig \in A.Beh.Sg \Rightarrow A'.Beh.St \in A.Beh.Sc$

Explanation of the rules is the following.

internal_action: the role-figure can change its capabilities and perform a state transition.

send_request: a role-figure may invoke a method in another role-figure by sending a method invocation request via the appropriate interface. The

[2] *RoleFigures* denotes all the role figure names in the configuration.

method signature must be declared in the type α of the target interface, and the arguments and return set in the request must match this signature.

recv_request: when receiving a request, the method signature must be declared in the type α of the interface, and sent arguments and return type must match this signature.

send_return, **recv_return:** when sending or receiving returns, there is no additional restrictions: only basic type compatibility check is made[4].

send_signal: a role-figure may send a signal to a role-figure due to service functionality. The signal must be declared in the type of the target interface.

recv_signal: receiving a signal will trigger a state transition.

Note that a state transition is allowed only during an internal action or when receiving a signal.

Adaptability functionality is dealt with six special requests: plug in *pi*, plug out *po*, create interface *ci*, behaviour change *bc*, capability change *cc*, and role-figure move *mo*. The corresponding rewriting rules are specialisations of **send_request** and **recv_request**, with specific constraints:

Role-figure Plug in: $\mathrm{A} \,\|\, pi \rightarrow \mathrm{A}' \,\|\, \boldsymbol{b}$ $\quad pi.arg ::= (name, loc, beh{:}\beta,\ cap{:}\pi)$
$\mathrm{A}.Int \subseteq \mathrm{A}'.Int \wedge \boldsymbol{b} = pi.arg.name \wedge \mathrm{location}(\boldsymbol{b}) = pi.arg.loc$
$\wedge\ \mathrm{B}.Beh = pi.arg.beh \ \wedge\ pi.arg.cap \subseteq \mathrm{B}.Cap$

Role-figure Plugout: $\mathrm{A} \,\|\, po \rightarrow \varnothing$ $\quad po.arg ::= (name)$
$\forall \mathrm{B},\ \mathrm{A}.Int \cap \mathrm{B}.Int \neq \varnothing{:}\ \mathrm{B} \rightarrow \mathrm{B}'$ with $\mathrm{B}'.Int = \mathrm{B}.Int - \mathrm{A}.Int$

Create Interface: $\mathrm{A} \,\|\, ci \rightarrow \mathrm{A}'$ $\quad ci.arg ::= (j_1 : \alpha_1, \ldots, j_n : \alpha_n)$
$\mathrm{A}'.Int = \mathrm{A}.Int \ \triangleleft_{i=1}^{n} ci.arg.j_i$

Behaviour Change: $\mathrm{A} \,\|\, bc \rightarrow \mathrm{A}'$ $\quad bc.arg ::= (beh : \beta,\ cSt{:}\ State)$
$\mathrm{A}.Beh.St \in \mathrm{A}.Beh.Ss \Rightarrow \mathrm{A}'.Beh.B = bc.arg.beh \wedge \mathrm{A}'.Beh.St = bc.arg.cSt$

Capability Change: $\mathrm{A} \,\|\, cc \rightarrow \mathrm{A}'$ $\quad cc.arg ::= (p_1{:}\ c_1, \ldots, p_n{:}\ c_n)$
$\mathrm{A}'.Cap = \mathrm{A}.Cap \ \triangleleft_{i=1}^{n} cc.arg.j_i$

Role Figure move: $\mathrm{A} \,\|\, mo \rightarrow \mathrm{A}'$ $\quad mo.arg ::= (loc)$
$\mathrm{A}'.Int \subseteq \mathrm{A}.Int \ \wedge\ \mathrm{A}'.Cap \subseteq \mathrm{A}.Cap \ \wedge\ \mathrm{location}(\boldsymbol{a}') = mo.arg.loc$

Role-figure Plug in: this method plugs in a new role-figure named *name* at location *loc*. The created role-figure ***b*** will also receive its behaviour *beh* and capabilities π. Its interfaces will be added to *A*, the role-figure that received the request. We hide the complex process of director play management, capability allocation, etc. and describe it by a single rewriting rule.

[4] Concerning the sending, the arguments are not checked because they have already been matched by the method invocation request semantics.

Role-figure Plug out: a role-figure which receives this request disappears. All references to its interfaces are removed with additional rewriting rules.
Create Interface: all the interfaces passed as arguments of this *ci* request will be added to the role-figure's interface definition, *Int.* Interface creation between two role-figures means that they will agree on the terms and conditions of their future interactions.
Behaviour Change: a behaviour change assigns a new behaviour to the role-figure, with a current state. It is allowed only in stable states A.*Beh.Ss*.
Capability Change: this request changes the capability definition of the role-figure. Capabilities specified in the *cc* request are added to the role-figure's capability set, *Cap.*
Role-figure Move: this request moves a role-figure from one location to another. It is equivalent to a sequence of *pi, bc ci, cc,* and *po* requests: a role-figure is plugged in at the new location *loc*, with the behaviour, interfaces and capabilities of the original role-figure. The role-figure instance at the original location is terminated by a *po* method.

5. ROLE-FIGURE PROPERTIES

In this section we introduce requirements on role-figure configuration, and define properties to verify their correctness. This verification process takes place at the service system design phase to improve the service system at early design phases by identifying design errors.

The role-figure configuration, $rfc= \{\boldsymbol{a}_1,..., \boldsymbol{a}_M, g_1...,g_N\}$, evolves through $rfc \rightarrow rfc^1 \rightarrow ... \ rfc^q \rightarrow ...$. In every transition a role-figure $\boldsymbol{a}_i$ evolves through A_i by either consuming a message g_j, generating a new message, or performing an internal action. Also, every interface in any of the role-figures is connected to another interface in another role-figure. The role-figure configuration is considered ***well-formed*** if and only if it obeys the rules and conditions constructed in the role-figure semantics. This role-figure configuration has three properties: *Plug ability*, *Consumption ability*, and *Play ability*.

Plug ability. This property proves that a role-figure has been plugged in at certain location. We do so by ensuring that the consumption of a plug in request has achieved the plug in of a role-figure at the appropriate location. The required capabilities and behaviour of the created role-figure must also satisfy the requirements of the plug in request. This property is defined by:

$$\boldsymbol{P}_{plug\ ability} = \quad \forall\ rfc=\{\mathbf{a}_1,\dots,\mathbf{a}_M,\mathbf{g}_1,\cdots,\mathbf{g}_N,\mathbf{g}_{plugin}\},\ \mathbf{g}_{plugin} = pi(\mathbf{a}_{new},loc_i,beh_i,capset_i),$$

$$rfc \xrightarrow{g_{plugin}} rfc' \parallel \mathbf{a}_{new}$$

$$\text{where} \begin{cases} rfc'=\{\mathbf{a}_1,\dots,\mathbf{a}_M,\mathbf{g}_1,\cdots,\mathbf{g}_N\} \\ \mathbf{A}_{\mathbf{a}_{new}} = <\mathbf{Beh} = <\mathbf{B} = pi.beh_i >, \mathbf{Cap} = pi.capset_i >; \mathbf{location(a}_{new}) = pi.loc_i \end{cases}$$

Consumption ability. This property proves that all messages generated by the role-figures during their execution will eventually be consumed:

$$P_{consumption\,ability} = \quad \forall\, rfc = \{\mathbf{a}_1, \ldots, \mathbf{a}_M, \mathbf{g}\}, \qquad rfc \xrightarrow{\mathbf{g}} rfc^1 \xrightarrow{\mathbf{g}_{rfc^1}} \cdots rfc^Q \xrightarrow{\mathbf{g}_{rfc^Q}} rfc_{\text{terminal}}$$

$$\text{such that} \begin{cases} \forall rfc^i: \quad rfc^i = \{\mathbf{a}_1, \ldots, \mathbf{a}_N, \mathbf{g}_1, \ldots, \mathbf{g}_P\}, & 1 < i < Q \\ rfc_{\text{terminal}} = \{\mathbf{a}_1, \ldots, \mathbf{a}_O\} \end{cases}$$

The configuration consumes messages and evolves based on the actions that will occur after the consumption: messages may be generated that will eventually be consumed. This process terminates when there will be no messages in the configuration (note the number of role-figures O in $rfc_{terminal}$ is different from M in rfc). This property examines all terminal states of a configuration and checks if they contain any unconsumed message (terminal states are states of the where no rewriting rule could be applied any further).

Play ability. This property proves that a role-figure, after its plug in phase, is playing or performing according to its predefined role. We have to verify that the role-figure behaviour is progressing, e.g. by marking certain states where something desirable happens as progress states, and examine if an execution of the system reaches such states. In play ability we only consider messages that are signals. There can be two types of this property: weak and strong Play ability. Weak Play ability proves that a role-figure has begun performing once it has been plugged in: at least one of the input signal $\boldsymbol{g_k}$ of the role-figure has been consumed. Weak Playability is defined by:

$$P_{wplayability} = \quad \forall\, rfc = \{\mathbf{a}_1, \ldots, \mathbf{a}_i, \ldots, \mathbf{a}_M, \mathbf{g}_1, \ldots, \mathbf{g}_N\}, \qquad rfc \longrightarrow \cdots rfc' \xrightarrow{\mathbf{g}_k} rfc''$$

$$\text{such that} \begin{cases} rfc' = \{\mathbf{a}_1, \ldots, \mathbf{a}_i, \ldots, \mathbf{a}_O, \mathbf{g}_1, \ldots, \mathbf{g}_k, \ldots, \mathbf{g}_P\}, \quad \mathbf{g}_k \in \mathbf{A}'_i.Beh_i.Sg_i \\ rfc'' = \{\mathbf{a}_1, \ldots, \mathbf{a}_i, \ldots, \mathbf{a}_O, \mathbf{g}_1, \ldots, \mathbf{g}_{k-1}, \mathbf{g}_{k+1}, \ldots, \mathbf{g}_P\} \end{cases}$$

Strong play ability requires that a role-figure is proved to be free of non-progress cycles (a progression is achieved):

$$P_{splayability} = \quad \forall\, rfc = \{\mathbf{a}_1, \ldots \mathbf{a}_i \ldots, \mathbf{a}_M, \mathbf{g}_1, \ldots, \mathbf{g}_N\} \quad rfc \longrightarrow \cdots rfc^q \xrightarrow{\mathbf{g}_{rfc^q}} \cdots rfc^{Q-1} \xrightarrow{\mathbf{g}_{rfc^{Q-1}}} rfc^Q$$

$$\text{such that} \begin{cases} rfc^q = \{\mathbf{a}_1, \ldots \mathbf{a}_i \ldots, \mathbf{a}_O, \mathbf{g}_1, \ldots, \mathbf{g}_P\}, & 1 \le q \le Q \\ \mathbf{A}_i \longrightarrow \mathbf{A}_i^1 \longrightarrow \cdots \mathbf{A}_i^q \rightarrow \cdots \mathbf{A}_i^Q, \\ \exists st_i \in \{\mathbf{A}_i^1.Beh_i.St_i, \cdots, \mathbf{A}_i^Q.Beh_i.St_i\}, \quad st_i \in \mathbf{a}_i|_{Beh.\Pr ogress} \end{cases}$$

The strong play ability requires knowledge of the role-figure state, which cannot be obtained by an external observation, as well as it requires knowledge of whether a state in the behaviour specification is a progress state or not. This property shows that a role-figure, which is assumed existing throughout a given execution of a configuration, evolves. Furthermore, the behaviour of the role-figure is said to have progressed at least once – one of its current states has been a progress state. The only difference to the weak play ability is the denotation, $\boldsymbol{a_i} \,|\, _{Beh.Progress}$, which stand for the progress states in the role-figure behaviour.

6. CONCLUSION

A formal model for the role-figures in the TAPAS architecture has been presented. Rewriting rules were used to describe role-figure behaviour as well as the three properties: *plug ability*, *consumption ability*, and *play ability*.

The plug ability property proves that a role-figure has been plugged in at certain location. The consumption ability property proves that all messages generated by the role-figures during their execution will eventually be consumed. The play ability property proves that a role-figure, after its plug in phase, is playing (i.e. its behaviour is progressing).

Although our experiences with modelling the role-figure and its properties are quite encouraging, the model we presented is just a first and preliminary step. The semantics and the dynamics of the role-figure model would benefit a more elaborated interface type theory: the behavioural types of Carrez et al.[12] can be used to describe the messages that are exchanged at the role-figure interfaces. Finally, the properties of the role-figure model may be extended to elaborate on the role-figure mobility management presented in [11], and used to verify the validity of mobility strategies (i.e. when and how to move).

References

1. F. A. Aagesen, B. E. Helvik, V. Wuvongse, H. Meling, R. Bræk, and U. Johansen, Towards a plug and play architecture for telecommunications, in *SmartNet'99* (1999)
2. F. A. Aagesen, B. E. Helvik, U. Johansen, and H. Meling, Plug&play for telecommunication Functionality: architecture and demonstration issues, in *IConIT'01* (May 2001)
3. F. A. Aagesen, B. E. Helvik, C. Anutariya, and M. M. Shiaa, On adaptable networking, in *ICT 2003* (April 2003)
4. R. Milner, J. Parrow, and D Walker, A calculus of mobile processes (parts I and II), in *Information and Computation*, 100:1-77 (1992)
5. E. Najm and J.B. Sstefani, A formal semantics for the ODP formal model, in *Computer Networks and ISDN systems 27*, pp.1305-1329 (1995)
6. J. Dustzadeh and E. Najm, Consistent semantics for ODP information and computational models, in *Proceedings of FORTE/PSTV'97* (Osaka, Japan, November 97)
7. N. Marti-Oliet and J. Meseguer, Rewriting logic as a logical and semantic framework, SRI International, Computer Science Laboratory Technical Report, August 1993
8. C. L. Talcott, An actor rewriting theory", in *ETCS*, 4 (1996)
9. G. Denker, J. Meseguer, and C. Talcote, Formal specification and analysis of active networks and communication protocols, in *DISCEX'2000* (January 2000)
10. B. Wang, J. Meseguer, and C. Gunter, Specification and formal analysis of PLAN algorithm in Maude, in *Workshop on Distributed system validation and verification*, (2000)
11. M. M. Shiaa, Mobility support framework in adaptable service architecture, in *Net-Con'2003* (Muscat Oman, October 2003)
12. C. Carrez, A. Fantechi, and E. Najm, Behavioural contracts for a sound assembly of components, in *Proc. of FORTE 2003, LNCS 2767* (Berlin, Germany, September 2003)

A LOCALIZED ARCHITECTURE FOR DETECTING DENIAL OF SERVICE (DoS) ATTACKS IN WIRELESS AD HOC NETWORKS

Mieso K. Denko
Department of Computing and Information Science,
University of Guelph, Guelph, Ontario, N1G 2W1

Abstract: In this paper we propose a reputation-based incentive scheme for detecting DoS attacks that target the network layer services. The scheme is based on clustering architecture to provide localized and scalable solutions. It involves a node history-based reputation update mechanism where more weights are given to the most recent reputation values. Load balancing feature was introduced to reduce the forwarding overhead on cooperative nodes. We evaluated the performance of the proposed scheme using simulation experiments. We studied a network with selfish nodes where the attack involves dropping packets. The effect of dropping control and data packets is investigated with and without load balancing. The results indicated that localized reputation-based incentive solutions can significantly increase packet delivery ratio in the presence of selfish nodes with limited communication and packet processing overheads.

Key words: Clustering, DoS attacks, MANET, Reputation, Wireless Networks

1. INTRODUCTION

Lack of cooperation in Mobile Ad Hoc Networks (MANETs) can occur due to misbehaving nodes or lack of sufficient resources. Enhancing cooperation among nodes in the network can help in detecting and mitigating DoS attacks caused by misbehaving nodes. Misbehaving nodes can be malicious or selfish. Selfish nodes are nodes that participate in the network to maximize their own benefit by using the resources of the network while saving their own resources. Malicious nodes directly attack the network by disrupting its normal operation. Existing incentive mechanisms for enforcing cooperation can be classified into trade-based [1,2,4] and reputation-based [3,5,6,7] mechanisms. While the former uses a payment-based incentive, the latter uses mutual ratings based on the services they provide to each other.

While extensive work has been carried out on confidentiality, integrity and privacy attacks [15], the threat to network availability has received less attention. Existing studies on Denial of Services (DoS) attacks concentrate on the analysis of various attack scenarios targeting a specific layer [16], or propose a probing mechanism to detect misbehaving node targeting a specific network layer function [14]. While using a probing mechanism can help in detecting DoS attacks, probing packets may introduce communication overhead in the larger network. Reputation rating coupled with localized probing mechanisms can alleviate the problem.

In this paper we propose a reputation-based incentive mechanism for detecting DoS attacks targeting packet dropping. We use a clustering architecture to provide a localized monitoring mechanism and enhance scalability. The main contributions of this paper are: (a) it provides a localized and scalable architecture for reputation management in a distributed manner (b) it provides a node history-based reputation maintenance mechanism which gives more weights to the recent reputation ratings and; (c) a load balancing mechanism was introduced to reduce the traffic on heavily used cooperative nodes.

The rest of this paper is organized as follows. Section 2 presents the description of the proposed scheme. Section 3 presents some optimization mechanisms to improve the reputation management. Section 4 provides the results of the performance evaluation. Finally, Section 5 presents conclusions and future work.

2. THE PROPOSED SCHEME

The DoS attacks can be active or passive. Active DoS attacks can modify the routing information or data packets, disrupt the network operation, or disable services by flooding the network or causing sleep-deprivation attacks. Active attacks on network routing include dropping packets, overloading routing traffic, routing table overflow and flooding. The passive DoS attacks do not alter the data but may result in packet dropping.

The two main schemes used in handling DoS attacks are detection and prevention. The detection scheme involves locating the attacker and taking appropriate actions. Monitoring nodes activity or tracing the attacker can help in detecting a DoS attack source. Several tracing and monitoring mechanisms have been proposed in the literature [8,9,17]. The prevention mechanism thwarts the DoS attacks before the attack is launched. It does so by identifying the attack packet and taking action before it reaches the target to be attacked. Common mechanisms used on the Internet include ingress or egress filtering and route-based packet-filtering mechanisms.

2.1 Assumptions and goals

A reputation based incentive mechanism was proposed for detecting the DoS attacks in MANETs. The mechanism motivates nodes to cooperate and detect DoS attacks caused by selfish nodes. It involves cluster formation, reputation data maintenance and the use of this information for DoS attacks detection and improving network performance. We make the following assumptions for the correct functioning of our scheme: (a) each mobile node has a unique ID and can join or leave the network freely. (b) each node knows its one-hop neighbors and operates in a promiscuous mode. (c) nodes are selfish but rational.

2.2 An Overview of the Proposed Scheme

Most existing reputation systems for MANETs [1,6,7], use global reputation computation and maintenance mechanisms. Since monitoring and detecting DoS attacks is a difficult task in a larger network, it is essential to design a mechanism that helps in reducing packet processing and communication overheads. A more suitable management strategy in this environment requires use of a distributed solution. A clustering architecture provides a distributed and scalable architecture for network monitoring, reputation data management and topology control. The localized and distributed feature also reduces the storage and communication overhead, thereby optimizing network performance [10].

Our proposal is based on the incentive mechanism presented in [12] and uses clustering architecture for localized reputation management. However, it can be built on top of any reputation system that uses localized control and management. The novelty of our scheme is the use of clustering to reduce the reputation data management overhead and improve the monitoring capability. The global reputation maintenance schemes may provide more data for decision-making, however, such schemes have several shortcomings. First, maintaining reputation data at every node congests the network by requiring each node to process multiple packets. Second, the exchanged information traverses multiple intermediate nodes and may be lost or altered. Third, such schemes require global synchronization and also incur high storage and communication overhead. Fourth, global reputation computation and maintenance is not scalable.

We considered two categories of selfish nodes, namely, non-selective selfish nodes (denoted as type 0) and selective selfish nodes. The non-selective selfish nodes drop both control and data packets. There are two types of selective selfish nodes denoted as type 1 and type2. The type 1 selfish nodes participate in forwarding control packets but drop the data packets. The type 2 selfish nodes forward data packets but do not participate in forwarding the control packets.

2.3 Election of the RM

Each cluster has a RM, multiple nodes and gateways. A RM is a node that is responsible for allowing inter-cluster communications and probing misbehaving nodes. For cluster formation, we use an aggregate index (*I*), which takes the node stability (*T*) and Reputation rating (*R*) into account. The value of *I* is computed as follows:

$$I = \alpha_1 T + \alpha_2 R, \text{ where } \alpha_1 + \alpha_2 = 1 \tag{1}$$

A node is eligible to become a RM only if it possesses the maximum aggregate index (*I*) compared to all its neighbors. A Hello message is used to maintain connectivity information. The node stability is determined by monitoring its cluster membership changes. Since reputation rating is one of the criteria used for electing the RM, the chance of electing a selfish node as RM is low.

2.4 Localized Reputation Data Management

The reputation data management process involves the development of strategies for the computation, storage and dissemination of reputation data. To distinguish between new and existing nodes, we maintain and exchange information about the node's age. This eliminates punishing recently-joined nodes that have not built their reputation yet. When a new node joins the network, an initial reputation value is assigned and the node's status is labeled as new. Its status will be monitored and its reputation ratings will be adjusted based on the service it provides.

2.4.1 Reputation computation and maintenance

Global detection of selfish nodes is a challenging task in MANETs, observing one-hop neighbors makes the management task easier. In this approach, nodes in each cluster monitor the behavior of their neighbors and update the reputation ratings. This is achieved by implementing the Watchdog mechanism [5] at each node. A watchdog mechanism detects non-forwarding nodes by overhearing packet transmission from neighbors. It requires continuous monitoring by relying on a promiscuous mode of operation. The reputation information is assigned and maintained as follows. Each node maintains the reputation of its neighbors locally and reports it to the RM periodically. Whenever a node–say A, gets service from node B, it rates the service by assigning (+1) for satisfactory service and by assigning (-1) for unsatisfactory service. The reputation rating is not exchanged among non-neighbors but is reported to the RM periodically.

However, before assigning a negative rating (-1), a node makes multiple forwarding trials. If no response is obtained, (-1) is assigned and a new node is used for packet forwarding. The threshold time (*k*) for the forwarding trial is determined based on node mobility, link failure or network load. The value of *k* would be longer in the presence of higher node mobility, link failure or network load. At node level, the reputation rate is updated based on the node's own information. However, when there is a tie, or when a suspicious node is encountered, it uses the reputation maintained at the RM and combines it with its local reputation. The reputation rating of node B at node A is computed as the difference between the total number of packets forwarded and the total number of packets dropped, divided by the total number of packets received by the forwarding node. It is scaled to lie between –100 percent and 100 percent. The threshold value is experimentally determined to decide beyond which value a node is considered selfish or cooperative. At the RM, the average of the reputation rating of a node is computed based on the node's neighbors' reputation information.

2.4.2 Packet probing at the RM

Distinguishing selfish nodes from congested nodes helps in avoiding the punishment of cooperative nodes with depleted resources. It also helps in finding alternative routes for packet forwarding until the nodes can recover from failure. Although the reputation ratings maintained at each node can be used to determine non-cooperation, it is not sufficient to distinguish between selfish and faulty nodes. We use a probe packet sent by the RM to the node's neighbors to distinguish selfish nodes from faulty nodes. The RM requests reputation data from each member of its cluster as part of the probing activity. We call a node faulty if it is unable to participate in the network services because of lack of sufficient resources due to reasons such as power outage, the node's current position in the network, and software fault. For this purpose, we use the probing packets generated by the RM. It is generated based on request from the nodes or periodically based on the status of received reputation ratings from nodes. The probe packet is sent to all neighbors of the suspected node. Upon probing, to avoid false accusations, the decision to warn or suspend a node is made only if at least 50 percent of the suspected node's neighbors report the misbehavior. A node with a warning status can be reinstated if it continues to cooperate.

Based on the probing results, a node that does not respond to all its neighbors is considered faulty, while a node that responds to only some nodes is considered selfish.

The actions taken after detecting faulty nodes are different from those taken against selfish nodes. Based on the information received from the desired nodes, the RM will issue a warning message or suspension from services. When a node is detected to be selfish, it will be warned and isolated temporarily or permanently. For faulty nodes, there will be no penalty leading to warning or permanent isolation, however, its reputation rating will be reset to the threshold value given to new incoming nodes. Routes via these nodes will then be temporarily unused until they recover.

3. OPTIMIZATION MECHANISMS

3.1 History-Based Reputation Updates Mechanism

The proposed incentive mechanism was built on top of a clustering architecture where nodes in each cluster collaborate in the detection of selfish nodes. Forwarding packets originated from cooperative nodes and refusing those generated from selfish nodes can motivate cooperation. Selfish nodes are isolated from the network only if they fail to cooperate after it's a period of warning.

To prevent a node from misbehaving after achieving a certain high-level reputation in the network, we assign weights, while updating the reputation ratings with more weights. The process gives more weight to recent values and less weight to past values. Let R_c and R_o be the current and the past reputation ratings respectively. Then, the updated reputation rating (R_u) is updated as follows:

$$R_u = \alpha R_c + (1 - \alpha)R_o \qquad (2)$$

Where α is a configurable parameter lying between 0.5 and 1. The values of R_c and R_o are computed as described in section 2.

3.2 Load Balancing for Cooperating Nodes

When a node issues a query or forwards a packet, it uses the reputation ratings to bias its decision towards forwarding data through more cooperative nodes. Each node normally forwards a packet via a node with a higher reputation rating. However, such a mechanism procedure may lead to overloading more cooperative nodes. Load balancing (LB) is one of the main issues that require attention among cooperative nodes that willingly forward packets to others. Load balancing enables distribution of the network load equally among all potential forwarding nodes. We have used randomization as a means of distributing the load among nodes with higher reputation ratings.

We have implemented a probabilistic packet forwarding strategy among eligible nodes based on their reputation ratings. In this strategy, the forwarding task is accomplished probabilistically by choosing the next hop among all candidate nodes. This helps in balancing the load within the networks while overcoming the effect of packet dropping and selective forwarding. The basic steps for the load balancing procedure are: First, the source node selects a set (*S*) of nodes from its neighbors with reputation ratings above a threshold value; next, the source node sends a packet to a randomly selected node from the set *S*; the process then continues until the packet reaches its destination.

4. PERFORMANCE EVALUATION

4.1 Performance Metrics

The effects of the fraction of selfish nodes, network size and simulation time on the performance were investigated using the following five metrics.

1. **Average packet delivery ratio.** Defined as the ratio of the total number of data packets received by destinations to the total number of packets sent by the source.
2. **Communication overhead.** Defined as the ratio of the total number of routing and reputation related packets transmitted to the total number of packets transmitted including data packets.
3. **Processing overhead.** Defined as the ratio of processing overhead introduced by reputation system to the total processing overhead including route computation and maintenance.
4. **Selfish node detection rate.** Defined as the ratio of the total number of selfish nodes detected to the total number of selfish nodes in the network.
5. **False-positive ratio.** Defined as the ratio of well-behaving nodes wrongly classified as selfish nodes to the total number of well-behaving nodes in the network.

4.2 Discussion of Simulation Results

We carried out simulation experiments using NS-2 [11] with mobile nodes roaming in a 1000m x 1000m square area with a transmission range of 250 m. The percentage of selfish nodes in the network lies between 0 percent and 50 percent. The selfish nodes were randomly selected among 50-200 mobile nodes. The random waypoint mobility model [13] was used with an average speed of 10 m/s and pause time of 50 seconds. The communication pattern uses 20 Constant Bit Rate (CBR) traffic with a data rate of four packets per second. The Ad Hoc On Demand Distance Vector (AODV) [18] protocol was used for routing.

The simulation results are shown in Figures 1 to 6. The data points in the graphs are based on the average of 20 simulation runs. Figure 1 shows the average packet delivery ratio with and without load balancing as a function of the fraction of selfish nodes. The delivery ratio decreases with the increase in the fraction of selfish nodes for both cases but with consistently better performance when load balancing is used. The results confirm that the use of the probabilistic forwarding mechanism reduces congestion at nodes with good reputations by increasing the packet forwarding and improving the packet delivery ratio. The simulation results in Figure 2 show that the selfish nodes detection rate increases from 91 percent to 99 percent with 40 percent selfish nodes and from 86 percent to 97 percent with 20 percent selfish nodes. When the fraction of selfish nodes increases in the network, the probability of detecting them increases. This is because such a node can be a neighbor to at least one node and can easily be detected by these neighbors. However, as the simulation time increases, the detection rates for both scenarios become similar.

Figure 3 shows that the false-positive ratio is between 2 percent and 4.5 percent when 20 percent of the nodes in the network are selfish whereas the ratio is between 2.3 percent and 5 percent when 40 percent of the nodes are selfish. This implies that misclassification increases relatively with both network size and fraction of selfish nodes. Cooperative nodes can be classified as selfish due to reasons such as packet loss caused by link failure or congestion. Mobility also results in misclassification of nodes. Figure 4 shows the simulation results of communication overhead as a function of the fraction of selfish nodes. The results indicate that the communication overhead increases slightly with an increase in the fraction of selfish nodes.

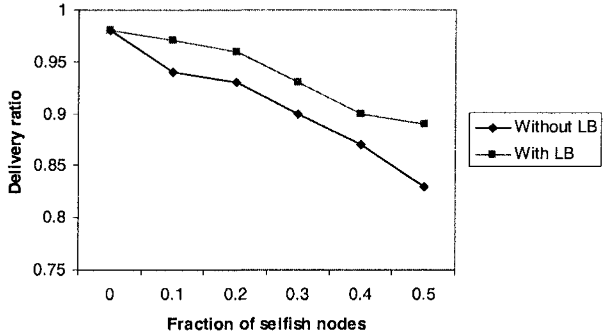


Figure 1. Packet delivery ratio with 100 mobile nodes

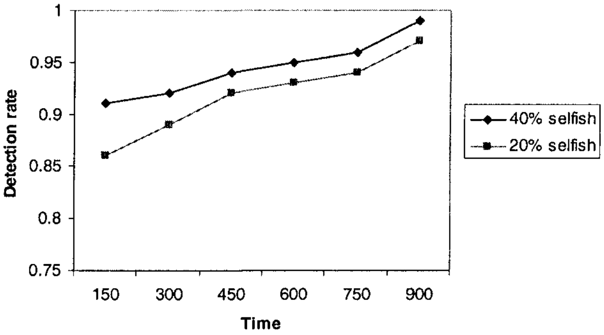


*Figure 2.*Detection rate with 100 mobile nodes

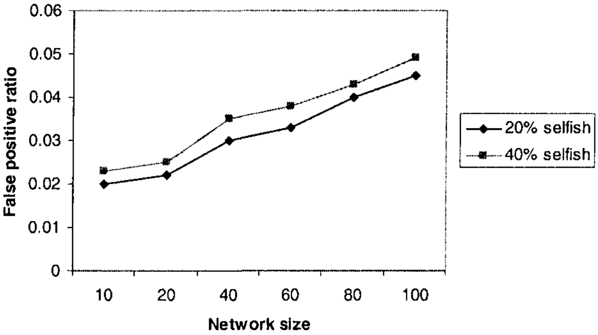


Figure 3. False-positive ratio with 100 mobile nodes

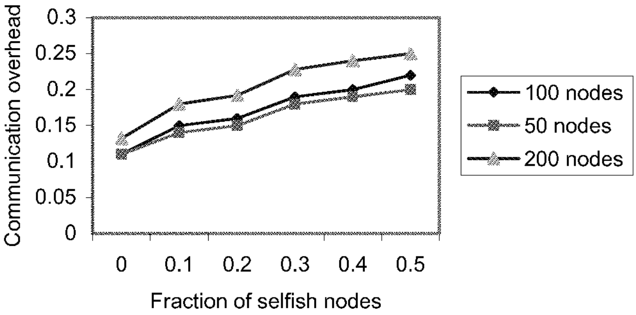


Figure 4. Communication overhead with 50, 100 and 200 mobile nodes

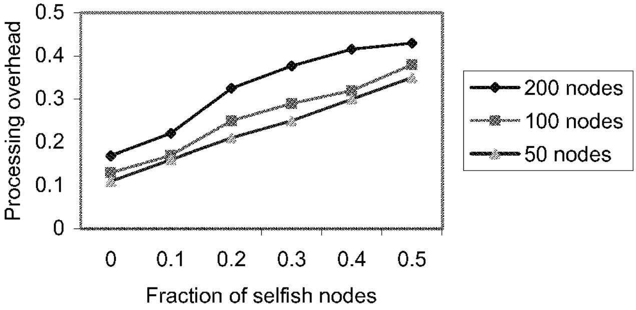


Figure 5. Processing overhead with 50, 100 and 200 mobile nodes

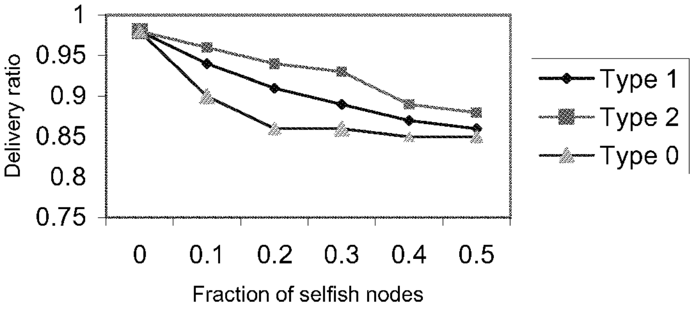


Figure 6. Packet delivery ratio with 200 mobile nodes

Little difference was observed, however, between the networks of size 50 and 100 nodes. The higher the percentage of selfish nodes, the slower the rate of increase, for larger network sizes. This implies that the use of clustering as a localized reputation data management scheme has introduced scalability and reduced communication overhead.

Figure 5 shows the results of packet processing overhead for three different network sizes. The packet-processing overhead increases slightly with an increase in the fraction of selfish nodes and network size. There is, however, a slight difference between the networks of size 50 and 100. The difference between the overheads caused by the simulated network sizes decreases slightly with an increase in the percentage of selfish nodes. The overall results indicate that the clustering architecture is effective in reducing the packet-processing overhead. Figure 6 shows the average packet delivery ratio for the three classes of selfish nodes as a function of the fraction of selfish nodes. The use of the probabilistic forwarding mechanism reduces congestion that could occur at cooperative nodes by introducing load balancing at each node. Both type 1 and type 2 selfish nodes have less effect on the delivery ratio than type 0 selfish nodes. However, the difference between the effects of the three classes of selfish nodes decreases slightly with an increase in the fraction of selfish nodes. This is partly due the possibility of direct communication between source and destination pairs. The little difference between the effects of type 0 and type 1 selfish nodes on packet delivery ratio suggests that the packet forwarding function is more crucial in improving the packet delivery ratio. Thus, a mechanism that enables selfish node to perform only the route request or reply operations correctly does not guarantee that the packet forwarding function will be properly performed.

5. CONCLUSIONS AND FUTURE WORK

In this paper we proposed a reputation-based incentive mechanism for detecting DoS attacks in MANETs. A clustering architecture was proposed for performing reputation data management in a localized and distributed manner. The node's reputation ratings and stability were taken into account for electing the RM. Load balancing mechanism was proposed to reduce the traffic on heavily used cooperative nodes. We have used the simulation technique to evaluate the network performance in the presence of selfish nodes. Our simulation results indicated that the reputation-based incentive mechanism is effective in tackling DoS attacks that occur due to selfish nodes. We will continue to investigate the performance of our incentive mechanism for tackling the DoS attacks by incorporating security mechanisms to improve network performance further. Our future work will also include comparisons of our scheme with existing similar schemes.

Reference

1. L. Buttyan, and J. Hubaux, Stimulating cooperation in self-organizing mobile ad hoc networks. ACM/Kluwer Mobile Networks and Applications (MONET) 8 (2003).
2. M. Baker, E. Fratkin, D. Guitierrez, T. Li, Y. Liu, and V. Vijayaraghavan, Participation incentives for ad hoc networks. http://www.stanford.edu/~yl31/adhoc (2001).
3. D. Barreto, Y. Liu, J. Pan, and F. Wang, Reputation-based participation enforcement for adhoc networks. http://www.stanford.edu/~yl314/adhoc (2002).
4. S. Zhong, J. Chen, and Y.R. Yang, Sprite - A simple, cheat-proof, credit-based system for mobile ad-hoc networks. Technical Report 1235, Department of Computer Science, Yale University (2002).
5. S. Marti, T.J. iuli, K. Lai, and M. Baker, Mitigating routing misbehavior in mobile ad hoc networks. In: Mobile Computing and Networking. (2000) 255–265.
6. S. Buchegger and J.Y.L Boudec, Performance Analysis of the CONFIDANT Protocol: Cooperation Of Nodes — Fairness In Distributed Ad-hoc NeTworks. In: Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC), Lausanne, IEEE (2002) 226–236.
7. P. Michiardi, and R. Molva, Making greed work in mobile ad hoc networks. Technical report, Institute Eur´ecom (2002).
8. A. Kuzmanovic, and E.W. Knight, Low-Rate TCP-Targeted Denial of Service Attacks. SIGCOMM'03, August 2003.
9. A.D. wood, and J.A. Stankovic, Denial of Service in Sensor Networks. IEEE October 2002.
10. W. R. Heinzelman, A.Chandrakasan, and H. Balakrishnan, Energy efficient communication protocol for wireless micro-sensor networks. Proceedings of IEEE Hawaii Int. Conf. on System Sciences, January 2000.
11. S. McCanne, and S. Floyd, Network Simulator. http://www.isi.edu/nsnam/ns/.
12. M.K. Denko: An Incentive-Based Service Differentiation in Mobile Ad Hoc Networks. IEEE International conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2005), August 2005, Montreal, Canada, to appear.
13. D.B. Johnson, and D.A. Maltz, Dynamic Source Routing in Ad hoc Wireless Networks'. In *Mobile computing* pages 153-181. Kluwer Academic Publishers,1996.
14. M. Just, E. Kranakis, and T. Wan, Resisting Malicious Packet Dropping in Wireless Ad Hoc Networks. Proc. of ADHOCNOW'03, Montreal, Canada.
15. I. Aad, J.P, Hubaux, and E.W. Knightly, Denial of Service Resilience in Ad Hoc Networks. ACM MOBICOM 2004, Philadelphia, PA, USA.
16. V. Gupta, S. Krishnamurthy, and M. Faloutsos, Denial of Service Attacks at the MAC Layer in Wireless Ad Hoc Networks. In Proc. of MILCOM, 2002.
17. A. Habib, M. H. Hafeeda, and B. Bhargava: Detecting Service Violation and DoS Attacks. Proc. of Network and Distributed System Security Symposium (NDSS), 2003.
18. C.E. Perkins, Ad hoc On-Demand Distance Vector (AODV) Routing, Internet Draft 17 February 2003.

A FUZZY PETRI NETS QOS MODEL FOR WIRELESS AD HOC NETWORKS

Lyes Khoukhi[1] and Soumaya Cherkaoui[1]

[1] Department of Electrical and Computer Engineering, Université de Sherbrooke J1K 2R1, QC, Canada

Abstract: In this paper, we explore the use of Fuzzy Petri Nets for QoS support in wireless ad hoc networks. We propose a fuzzy Petri nets technique for modeling and analyzing the QoS decision making for traffic regulation. The proposed model, called FPWIM, studies the fuzzy regulation traffic rules in order to deal with the imprecise information caused by the dynamic topology of ad hoc networks. The input parameters of FPWIM are the node mobility and the delay measurement received by a node as feedback information from the MAC layer. The output parameter of FPWIM is the traffic regulation rate necessary to avoid the possible congestion in the network. Different traffic and network motilities are considered by FPWIM in order to help make an efficient QoS decision for various network conditions.

Keywords: ad hoc networks; quality of service; fuzzy Petri networks.

1. INTRODUCTION

As a widespread use of wireless technology, the ability of mobile wireless ad hoc networks to support real-time services with Quality of Service (QoS) has become a challenging research subject. This challenge is due essentially to the fact that the wireless topology can change rapidly in unpredictable ways or remain relatively static over long periods of time. In addition, the dynamic topology of ad hoc networks generates imprecise and uncertain information, which may complicate the task of QoS and routing protocols.

Some researches have focused on aspects such as QoS routing [7]-[8] and MAC layer [9]. Other recent researches have focused on presenting models that enable QoS support independent of the routing protocols. The most noteworthy QoS models attempting to establish comprehensive solutions for MANETs (Mobile Ad hoc Networks) are INSINGIA [2], SWAN [3], and FQMM [4]. We have proposed in [5], an intelligent QoS model with service differentiation based on neural networks in mobile ad hoc networks named GQOS. In [6], we have developed the FuzzyMARS model which explores the use of a fuzzy logic semi-stateless QoS approach comparatively to using neural networks. The use of fuzzy logic showed very interesting results such as the reduction of the average end-to-end delay of traffic. This study aims at giving a good analytical model for using fuzzy logic for traffic regulation in MANETs.

The chosen model is fuzzy Petri Nets. The classical Petri nets [15] are not sufficient to model the dynamic topology of MANETs characterized by the uncertainty and imprecision information. It has been proven that the imprecise information can be represented efficiently by using Fuzzy Petri Nets model [10] [11] [12] [13].

The proposed fuzzy modeling scheme for traffic regulation aims to represent the dynamic adjustment of traffic transmission according to the network conditions. We called this model "FPWIM". FPWIM exploits the fuzzy concepts to model the QoS approach decision making. The representation of different fuzzy processes for decision making can be performed by formulating the production rules of these processes. Each fuzzy production rule is a set of antecedent input conditions and consequent output propositions. We proceed to construct the previous aspects (the input and output parameters) of the production rules in order to better represent and understand the process of traffic regulation in wireless ad hoc networks. The traffic regulation used to avoid the congestion depends on the traffic state and the dynamic topology of the network. The input parameters of FPWIM are the node mobility and the delay-measurement. This later parameter is received by a node as feedback information from the MAC layer; it represents the time taken by packet to reach the destination. The delay measurement parameter can give information about the status of a network in terms of congestion. A big value of this parameter signifies that congestion may have appeared in the network. Therefore, the process of traffic regulation should be started. The amount of this regulation represents the output parameter of FPWIM. The fuzzy Petri nets tool is used for its efficiency and flexibility over other modeling tools in the aim of better modeling and representation the process of traffic regulation.

This paper is organized as follows: in Section II, we describe the fuzzy Petri nets tool. Section III illustrates the fuzzy regulation traffic rules used by FPWIM. The fuzzy Petri nets model for traffic regulation is shown in Section IV. Finally, Section IV concludes the paper.

2. FUZZY PETRI NETS

Classical Petri Nets [15] do not have sufficient capacity to model the uncertainty in systems [14] [18]. This limitation of Petri nets has encouraged researchers to extend the exiting models by using the fuzzy reasoning theory [10] [11] [13]. The combination of Petri nets models and fuzzy theory has given rise to a new modeling tool called Fuzzy Petri Nets (FPN). FPN formalism has been widely applied in several applications such as, fuzzy reasoning systems [16], robotics systems [12], and real-time control system [14], etc.

In what follows, we give a brief description about the FPN modeling tool [10] [12]. Let consider FPN = (PN, CND, MF, FSR, FM).

The tuple PN = (P, T, A, FW, FH) is called Petri nets if: (P, T, A) is a finite net, where [14]:

P = {P_1, P_2, ..., P_n} is a finite non-empty set of places,

T = {T_1, T_2, ..., T_n} is a finite non-empty set of transitions,

$A \subseteq (P \times T) \cup (T \times P)$ is a finite set of arcs between the places and transitions or vice versa.

FW: A→ N^+ represents a weighting function that associates with each arc of PN a non-negative integer of N^+.

$FH \subset (P \times T)$: represents an inhibition function that associates a place $P_i \in P$ contained in FH (T_j) to a transition T_j itself.

a) CND = {cd_1, cd_2,..., cdn} represents a set of conditions that will be mapped into the set P; each $cd_i \in CND$ is considered as one input to the place $P_i \in P$. A condition cd_i takes the form of "X is Z", which means a combination between the fuzzy set Z and the attribute X of the condition. For instance, in the condition "the delay measurement is small", the attribute "X = delay measurement" is associated to the fuzzy set "Z= small", but other fuzzy sets can also be considered (e.g. "Z = medium", "Z=large", etc.).

b) Consider MF: $u_Z(x) \rightarrow [0,1]$, a membership function which maps the elements of X (as defined in b.) into the values of the range [0,1]. These values represent the membership degree in the fuzzy set Z. The element x belonging to X represents the input parameter of the condition "X is Z", and $u_z(x)$ measures the degree of truth of this condition. Note that the composition of membership function degrees of the required conditions is performed by fuzzy operators such as MIN/MAX.

c) Let consider the following rule R_i: "R_i: if x_1 is z_1 and /or x_2 is z_2 then A is B". The firing strength function of rule R_i (FSR_i) represents the strength of belief in R_i. The conclusion of R_i (modeled by CSR_i) can take one of the following forms:

$$CSR_i = MIN(u_{z1}(x_1), u_{z2}(x_2)) = u_{z1}(x_1) \wedge u_{z2}(x_2)$$

$$CSR_i = MAX(u_{z1}(x_1), u_{z2}(x_2)) = u_{z1}(x_1) \vee u_{z2}(x_2)$$

d) SWR is the selected wining rule R_L among the n-rules R_1, R_2, ..., R_n. SWR is the rule which has the highest degree of truth. Let FSR_L be the corresponding firing strength of R_L, then the selected rule SWR is given as follows:

$$SWR = MAX(FSR_1, FSR_2, ..., FSR_n)$$

e) The marking task in FPN illustrates the satisfaction of events occurred during the performance of fuzzy rules. This marking function called "fuzzy marking" (FM) distributes the tokens over the places of the nets. A token is the primitive concept used in classical Petri nets for the definition of their execution.

f) The sequence $\delta = \langle T_1, T_2, ..., T_n \rangle$ is said to be reachable from a fuzzy marking FM$_1$, if $T_i \in T$ is a firable from FM$_{i-1}$ $\in$ FM and leads to FM$_{i+1}$ $\in$ FM, for all transitions $T_i \in \delta$. The firing of transition $T_i \in T$ (Figure 1) is performed in two steps: a) T_i removes tokens and then, b) T_i places tokens.

3. FUZZY REGULATION TRAFFIC RULES USAGE

Most of fuzzy systems use the following form for modeling [1] [19] [17]: Rule R: if Ip$_1$ is A AND Ip$_2$ is B then Op is C
Where:

- Ip$_1$ and Ip$_2$ are the input parameters,
- Op is an output parameter,
- A, B, and C are fuzzy sets,
- AND represent fuzzy operator,
- The fuzzy conditions of rule R are "Ip$_1$ is A", and "Ip$_2$ is B".

The construction of the above aspects (inputs, outputs, and fuzzy sets) for performing the traffic regulation to avoid the possible congestion depends on the traffic state and the dynamic topology of wireless ad

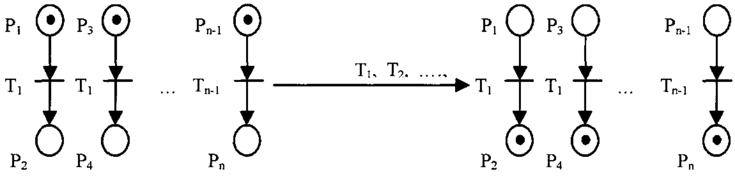


Figure 1. The transitions firing in FPN

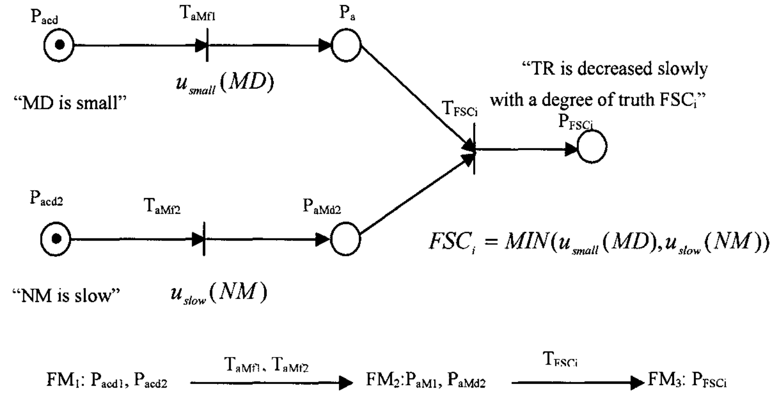


Figure 2. The modeling of fuzzy rules structure and its dynamic behaviour

hoc networks. Thus, the previous fuzzy aspects can take various values:

- The first input parameter: is represented by the Delay-Measurement (DM) at a mobile node. DM can be either small or large.
- The second input parameter: is represented by the Node Mobility (NM). NM can either be slow or medium (note that "fast node mobility" is included in the case of "medium node mobility").
- The output parameter: is represented by the Traffic regulation rate (TR). TR can either be decreased or increased (slowly or largely).

The aim is to help to establish production rules that make an efficient QoS decision. In the following, we explain the proposed fuzzy tool for the QoS decision making. Let consider the following fuzzy rule R_L:

Rule R_L: if DM is small and NM is slow, then TR is increased largely.

R_L takes into consideration the input parameter of the feedback delay-measurement DM and the node mobility NM. The traffic regulation rate TR represents the output parameter. Figure 2 illustrates FPN that models the dynamic aspect of the fuzzy rule R_L.

- P_{acd1}: models the antecedent condition 1 (acd_1) of R_L; acd_1 = "DM is small".
- P_{acd2}: models the antecedent condition 2 (acd_2) of R_L; acd_2 = "NM is slow".
- T_{aMf1}: models the membership function of the antecedent condition 1; $T_{aMf1=}u_{small}(DM)$.
- T_{aMf2}: models the membership function of the antecedent condition 2; $T_{aMf2=}u_{slow}(NM)$.
- P_{aMd1}: models the membership degree value of the condition 1 of a rule R_L. This value determines the satisfaction degree of the DM input parameter to the fuzzy set "small".
- P_{aMD2}: models the membership degree value of the condition 2 of a rule R_L. This value determines the satisfaction degree of the NM input parameter to the fuzzy set "slow".
- T_{FSCL}: models the operation of minimum composition "MIN" between the antecedent conditions (e.g. condition 1 and condition 2) of a rule R_L. The firing strength of R_L is represented by the MIN operation: $MIN(u_{small}(DM), u_{slow}(NM))$.
- P_{FSCL}: models the value of the firing strength of R_L. This value defines the degree of truth of the output proposition "TR is increased largely".

4. FUZZY PETRI NETS MODEL FOR TRAFFIC REGULATION

We consider the following rules:

R_1: if DM is small and NM is slow then TR is increased largely,
R_2: if DM is small and NM is medium then TR is increased,
R_3: if DM is large and NM is slow then TR is decreased,

R_4: if DM is large and NM is medium then TR is decreased largely.

- Input parameters: The input parameter of the first antecedent condition of the rules R_1, R_2, R_3, and R_4 is the delay measurement DM. The input parameter of the second antecedent condition of the rules R_1, R_2, R_3, and R_4 is the node mobility NM.
- Fuzzy sets: The fuzzy set of the antecedent conditions of the defined rules R_1, R_2, R_3, and R_4 are: small, large, slow, and medium.
- Antecedent conditions (acd_i): The first antecedent condition (acd_1) in the rules R_1, R_2, R_3, and R_4 is: acd1: DM is small; acd2: DM is large. The second antecedent condition (acd_2) in the rules R_1, R_2, R_3, and R_4 is: acd1: NM is slow; acd2: NM is medium.
- Output parameters: The output parameter of the rules R_1, R_2, R_3, and R_4 is the traffic regulation rate TR.
- The decisions making of the rules R_1, R_2, R_3, and R_4 are: increased largely, increased, decreased, decreased largely,
- The fuzzy logic operator used by the rules R_1, R_2, R_3, and R_4 is AND.

The fuzzy operator "AND" is used to combine the two antecedent conditions of each rule using the MIN function. This provides the firing strength value for each rule. After that, MAX composition function is used to combine all firing strength values of the defined rules R_1, R_2, R_3, and R_4 in the aim of determining the highest one that will be the selected wining rule. Figure 3 shows the fuzzy logic scheme for decision making of rules R_1, R_2, R_3, and R_4.

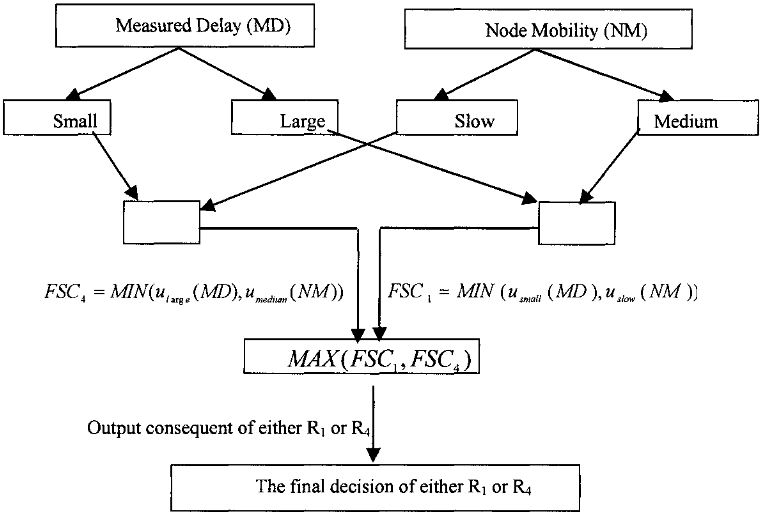
In what follows, we illustrate the steps of the proposed FPN model.

a) Enter the input parameters into the places and transitions:

- $P_{IP} = \{P_{IP1}, P_{IP2},...., P_{IPn}\}$ is a set of places representing the input parameters. In the Figure 4, P_1 and P_2 represent respectively, the first (e.g. delay measurement DM) and second (e.g. node mobility NM) antecedent condition of the rules R_1, R_2, R_3, and R_4.
- $T_{IP} = \{T_{IP1}, T_{IP2}, ..., T_{IPn}\}$ represents a set of input parameter transitions. The transitions T_{IP1} and T_{IP2} illustrated in Figure 4 are used to distribute respectively, the input parameters "DM" and "NM" for making the first and second antecedent conditions of the defined rules R_1, R_2, R_3, and R_4.

b) Represent the antecedent conditions, and compute the membership function for each condition.

- $P_{acd}= \{P_{acd1}, P_{acd2}, ..., P_{acdn}\}$ is a set of places that represent the antecedent conditions. P_{acd1} and P_{acd2} in the model presented in Figure 4 describe respectively, the antecedent conditions "acd_1" and "acd_2".
- $T_{aMf} = \{T_{aMf1}, T_{aMf2},...., T_{aMfn}\}$ is a set of transitions that represent the antecedent membership functions. T_{aMf1}, T_{aMf2}, T_{aMf3}, T_{aMf4} observed in Figure 4 represent the membership functions of respectively, $u_{small}(DM)$, $u_{l\arg e}(DM)$, $u_{slow}(NM)$, $u_{medium}(NM)$.
- $P_{aMd} = \{P_{aMd1}, P_{aMd2},..., P_{aMdn}\}$ is a set of places that represent the antecedent membership degrees. The values of the place P_{aMd1} indi-


Measured Delay (MD)
Node Mobility (NM)
Small
Large
Slow
Medium
$FSC_4 = MIN(u_{large}(MD), u_{medium}(NM))$
$FSC_1 = MIN(u_{small}(MD), u_{slow}(NM))$
$MAX(FSC_1, FSC_4)$
Output consequent of either R_1 or R_4
The final decision of either R_1 or R_4

Decision making algorithm:

- Phase 1: enter the input parameters of the rules R_1, R_2, R_3, and R_4.
- Phase 2: calculate the degree of truth of the antecedent conditions.
- Phase 3: apply the operation of minimum composition (MIN) with the fuzzy operator AND/OR in order to generate the firing strength value for each rule R_1, R_2, R_3, and R_4.
- Phase 4: apply the operation of maximum composition to select the wining rule among the rules R_1, R_2, R_3, and R_4.
- Phase 5: generate the output consequent of the selected wining rule.

Figure 3. The fuzzy decision making mechanism of FPWIM

cates the degree of satisfaction of the input parameter DM to the fuzzy set "small".

c) Compute the firing strength of conditions

- T_{FSC} = {T_{FSC1}, T_{FSC2}, ..., T_{FSCn}} represent a set of transitions that model firing strength conditions. For instance, the transition T_{FSC1} shown in Figure 4 performs the operation of minimum composition (MIN) on the antecedent conditions of the rule R_1: $MIN(u_{small}(MD), u_{slow}(NM))$. Note that the fuzzy operator AND is integrated with the MIN operation to combine the first and second conditions of R_1.
- P_{FSC}= {P_{FSC1}, P_{FSC2}, ..., P_{FSCn}} is a set of places that represent the firing strength. P_{FSCi} tokens are proportional to the number of antecedent conditions of a rule R_i. This number is shown by the label illustrated between the transitions T_{aMfi} and the place P_{aMdi}. The construction of the antecedent conditions of a rule R_i is performed by firing a transition T_{FSCi}. The inhibitor arc designed between a place P_{FSCi} and T_{FSCi} is useful to note that T_{FSCi} should fire one time.

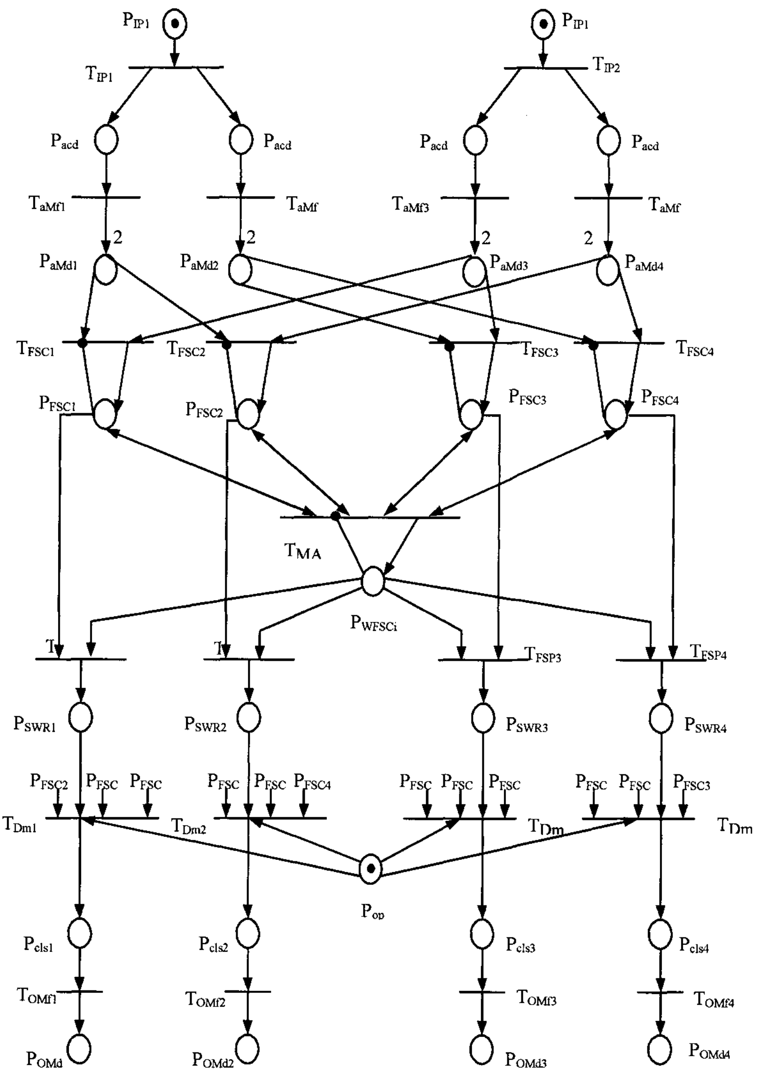


Figure 4. FPWIM model

d) Determine the selected wining rule among the activated rules:

- T_{FMAX} = MAX $\{P_{FSC1}, P_{FSC2}, \ldots, P_{FSCn}\}$ is a transition that models the maximum composition operation (MAX) for the defined rules. The firing strength value of a rule R_i is stored in the place P_{FSCi}.
- P_{WFSCi} represents the firing strengh condition FSC_i of the selected wining rule R_i. The later rule is determined as in the following step.
- $T_{FSP} = \{T_{FSP1}, T_{FSP2}, \ldots, T_{FSPn}\}$ is a set of transitions that model the firing strength comparison. For instance, the transition T_{FSP3} is useful

to make a comparison between FSC_3 of the rule R_3 and the selected wining firing strength $WFSC_i$.

- $P_{SWR} = \{P_{SWR1}, P_{SWR2}, \ldots, P_{SWRn}\}$ is a set of places that models the selected wining rules. The rule R_i is selected to be fired if the place P_{SWRi} contains a token.

e) The conclusion of the selected rules:

- $T_{Dm} = \{T_{Dm1}, T_{Dm2}, \ldots, T_{Dmn}\}$ is a set of transitions that represent the decision of the selected rule. T_{Dmi} deletes the firing strength values of other rules in order to fire only the selected rule R_i.
- P_{op} is a place that models the output parameter. As shown in Figure 4, the place P_{op} represents the traffic regulation rate TR.
- $P_{cls} = \{P_{cls1}, P_{cls2}, \ldots, P_{clsn}\}$ models a set of places that describe the different decisions of the defined rules. The places P_{cls1}, P_{cls2}, P_{cls3}, and P_{cls4} illustrate the following conclusions respectively, "increased largely", "increased", "decreased", and "decreased largely". Only one place among all places will contain a token which represent the conclusion of the selected wining rule. For instance, the conclusion of the selected rule R_1 is "increased largely" if T_{Dm1} transfers a token from the place P_{SWR1} to the place P_{cls1}.
- $T_{OMf} = \{T_{OMf1}, T_{OMf2}, \ldots, T_{OMfn}\}$ is a set of transitions that represent the output membership functions. T_{OMf1}, T_{OMf2}, T_{OMf3}, and T_{OMf4} represent the calculation performed by the used fuzzy method to compute the membership degree of respectively, $u_{l\arg e\_increase}(TR)$, $u_{increase}(TR)$, $u_{decrease}(TR)$, $u_{l\arg e\_decrease}(TR)$,
- $P_{OMd} = \{P_{OMd1}, P_{OMd2}, \ldots, P_{OMdn}\}$ is a set of places that represent output membership degree. The places P_{OMd1}, P_{OMd2}, P_{OMd3}, and P_{OMd4} indicate that the output parameters of "TR is increased", "TR is increased largely", "TR is decreased", and "TR is decreased largely" are satisfied with the following membership degree, $u_{l\arg e\_increase}(TR)$, $u_{increase}(TR)$, $u_{decrease}(TR)$, $u_{l\arg e\_decrease}(TR)$, respectively.

5. CONCLUSION

In this paper, we proposed FPWIM which is a fuzzy Petri nets technique for modeling and analyzing the QoS decision making for traffic regulation in wireless ad hoc networks. We examined the fuzzy production rules used for traffic regulation process by identifying the different parameters of each rule. The input parameters of FPWIM rules are the node mobility and the delay measurement received by a node as feedback information from the MAC layer. The output parameter of FPWIM rules is the traffic regulation rate required for reducing the possible congestion occurred in the network. The established fuzzy production rules will help deal with changing network situations in terms of mobility and congestion. This allows making an efficient QoS decision for different variable network conditions.

REFERENCES

1. Zadeh, L. A., Knowledge representation in fuzzy logic, *IEEE Transaction knowledge Data Engineering*, 1, 89-100, 1989.
2. Lee, S.-B., Ahn, G.-S., Zhang, X., and Campbell, A.T., INSIGNIA: An IP-Based Quality of Service Framework for Mobile Ad Hoc Networks, *Journal of Parallel and Distributed Computing, special issue on wireless and mobile computing and communication*, vol. 60, no. 4, pp. 374-406, Apr. 2000.
3. Ahn, G.H., Campbell, A. T., Veres, A., and Sun, L. H., SWAN: Service Differentiation in Stateless Wireless Ad Hoc Networks, *IEEE INFOCOM* 2002.
4. Xiao, H., Seah, W. K.G., Lo, A., and Chaing, K., Flexible QoS Model for Mobile Ad-hoc Networks, *In the Proceedings of IEEE Vehicular Technology Conference*, vol. 1, pp 445-449, Tokyo, May 2000.
5. Khoukhi, L., Cherkaoui, S., A Quality of Service Approach Based on Neural Networks for Mobile Ad hoc Networks, *IEEE-IFIP International Conference on Wireless and Optical Communications Networks*, WOCN 2005, Dubai, UAE, Mar. 2005.
6. Khoukhi, L., Cherkaoui, S., FuzzyMARS, A Fuzzy Logic Approach with Service Differentiation for Wireless Ad hoc Networks, *IEEE International Conference on Wireless Networks, Communications, and Mobile Computing* WirelessCom2005, June 13-16, 2005.
7. Khoukhi, L., Cherkaoui, S., Flexible QoS Routing Protocol for Mobile Ad Hoc Networks, *In the Proceedings of the 11th IEEE International Conference on Telecommunication* (ICT2004), Brazil, Aug. 2004.
8. Lin, C. R., and Liu, J.-S., QoS Routing in Ad Hoc Wireless Networks, *IEEE Journal on Selected Areas in Communication*, vol. 17, no. 8, 1426–1438, 1999.
9. Lin, C.-R.: On-Demand QoS Routing in Multihop Mobile Networks, *IEEE INFOCOM* 2001, pp. 1735–1744, April 2001.
10. Ashon, S.I., Petri net models of fuzzy neural networks, *IEEE Transaction on System, Man, and Cybernetics*, 25, 926-932, 1995.
11. Chen, S.-M., Ke, J.-S., and Chang, J.-F., knowledge representation using fuzzy Petri nets, *IEEE Transaction on Knowledge Data Engineering*, 2, 311-319, 1990.
12. Cao, T., Sanderson, A.C., Variable reasoning and analysis about uncertainty with fuzzy Petri nets, *In the proceeding of 14 th International conference on application and theory of Petri nets*, Troy, NY, August, pp. 126-175.
13. Looney, G., Fuzzy Petri nets for rule-based decision making, *IEEE Transaction on System, Man, and Cybernetics*, 18, 178-183, 1998.
14. Murata, T., Suzuki, T., and Shatz, S. M., Fuzzy-timing high-level Petri net model of a real-time network protocol, *In the proceeding of ITC-CSCC 96*, Seoul, Korea, July 15-17, pp. 1170-1173.
15. Dwyer, M. B., and Clarke, L. A., A compact Petri net representation and its implication for analysis, *IEEE Transaction Software Engineering*, 22, 794-811, 1996.
16. Chaudhury, A., Marinescu, D. C., and Whinston, A., Net-based computational models of knowledge processing systems, *IEEE Expert*, 8, 79-86, 1993.
17. Eshera, M. A., and Barash, S. C., Parallel rule-based fuzzy inference on mesh-connected systolic array, *IEEE Expert*, winter, 4, 17-35, 1989.
18. Murata, T., Temporal uncertainty and fuzzy-timing high level Petri nets, *In the proceeding of 17th International Conference of Application and Theory of PNs*, Osaka, Japan, June 26, pp. 11-28, 1996.
19. Polat, F., and Guvenir, H., UVT: A unification-based tool for knowledge base verification, *IEEE Expert*, 8, 69-75, 1993.

USING MOBILE AGENT FOR LOCATION-SPECIFIC DATA RETRIEVAL IN MANET

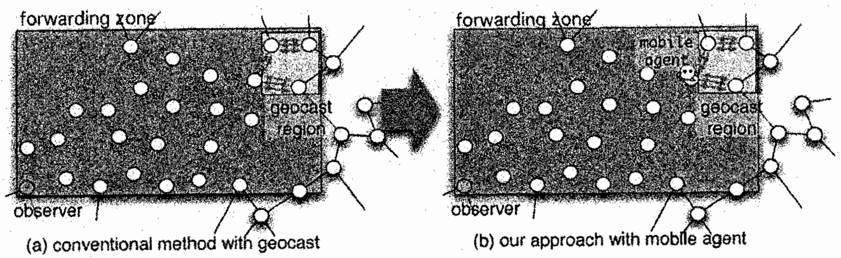
Kenji Tei[1,2], Nobukazu Yoshioka[2], Yoshiaki Fukazawa[1], and Shinichi Honiden[2,3]
[1]*Waseda University, 3-4-1 Okubo, Shinjuku-ku, Tokyo, 169-8555 Japan;* [2]*National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430 Japan;* [3]*The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656 Japan*

Abstract Location-specific data retrieval is an attractive application in a Mobile Ad-hoc Network (MANET). Simple solution for it is that an observer retrieves the data by geocasting from an observer node, but its overhead highly depends on location of the observer and the designated region. We propose a mobile agent approach. A mobile agent migrates from the observer node to a node in the designated region, retrieves data from there, and summarizes, filters, and compresses the retrieved data, This data is sent back to the observer, when the observer request. Since the data is retrieved by the mobile agent located near the data sources, the data retrieval in the mobile agent approach would involve low overhead, even if the observer is far from the target region or moves around. In the MANET, however, even after the first migration, to stay near data sources, a mobile agent should migrate to another node in response to node movements.. In this paper, we propose the Geographically Bound Mobile Agent (GBMA) which is a mobile agent that migrates to always be located in a designated region. Moreover, to clarify where the GBMA should be located and when the GBMA starts to migrate, we introduce two geographic zones: required zone and expected zone. Compared with the conventional methods with geocast or with a conventional mobile agent, the GBMA with these zones for retrieving location-specific data can reduce the total number of messages.

Keywords: location-specific data retrieval, mobile ad-hoc network, mobile agent

1. Introduction

Mobile Ad-hoc Networks (MANETs) consisting of mobile wireless nodes that communicate with each other have been receiving great attention[1]. The MANET without a fixed infrastructure is expected to be effective in post disaster areas where fixed infrastructures have been destroyed[2}. We believe that people under such situation become more voluntary and cooperative than usual, and that they will provide their personal mobile devices, such as PDAs or smart phones, to create and to maintain the MANET used for communicating



(a) conventional method with geocast (b) our approach with mobile agent

Figure 1. Applying mobile agent in a MANET

with each other. In the post disaster area, location-specific data in particular is exchanged. For example, an observer may confirm a safety of missing people among patients in a certain hospital, locate disaster victims in a certain dangerous place, or inquire what goods are necessary at a certain shelter. We call the region in which the observer is interested *designated region* in this paper.

Simple approach to the location-specific data retrieval is that an observer sends, from the observer node, messages to retrieve this data. Geocast based on directed flooding[4][5][6] can make this approach produce low message overhead. Geocast[3] is a subclass of multicast with the multicast group defined by a geographic region, and is the delivery of messages to nodes within this region. Geocast based on directed flooding, such as LBM[4], Voronoi-based routing[5], or GeoGRID[6], is a subclass of geocast whose messages are delivered by directed flooding. One of major protocols in them is LBM. LBM adopts directed flooding and defines forwarding zone. When receiving a LBM message, only a node in a forwarding zone described in this message forwards this message, and otherwise discards it. Because nodes in the forwarding zone is subset of whole nodes in a MANET, LBM reduces the message flooding overhead. Adopting LBM for the message delivery, the overhead of this approach can be reduced. We call this approach adopting LBM geocast *simple geocast approach* in this paper.

However, overhead of the simple geocast approach highly depends on the location of an observer and a designated region, even if it adopts LBM. Consider the case that the observer tries to survey the necessary goods at certain shelters. The observer is sometimes located far from the region, and exchanges data from there, as shown in Figure 1 (a). The circles represent nodes and the arrows represent message transfer. The observer (the circle at lower left) frequently interacts with many evacuees in the shelter (the rectangle at top right). Then, data are exchanged many times between the observer and the evacuees. Moreover, one data exchange involves many message forwardings, since forwarding zone covers a large region according to a distance between the observer and the shelter. As a result, for the observer far from its designated region or moving around, the simple geocast approach will produce the massive number of messages in spite of limiting the number of forwarding nodes.

In this paper, we propose a mobile agent approach[7] for the location-specific data retrieval.In client/server applications, the use of the mobile agent migrating from a client node to a server node and interacting locally with the server can reduce network traffic[8]. We apply this scheme in a MANET. In our method, a mobile agent migrates from an observer node to an evacuee node in a designated region, and retrieves data from evacuee nodes in this region (Figure 1 (b)). After this migration, the mobile agent can retrieve the data via shorter length routes with a smaller number of messages. Therefore, the number of messages involved in the mobile agent approach would not highly depend on location of the observer and the designated region.

However, conventional mobile agent may not provide sufficient solution for the data retrieval. Because each node in a MANET can physically move, the mobile agent host node may move apart from the designated region during the data retrieval. This increases of data retrieval overhead of the mobile agent and makes tracking of the mobile agent difficult. In response to node movements, the mobile agent should migrate among nodes while retrieving data, to remain in the designated region. We propose a mobile agent reactively migrating to remain the designated region in response to the host node movements, and name it *Geographically Bound Mobile Agent* (GBMA). This GBMA will provide a solution for the location-specific data retrieval.

This section has presented the background of our work. The rest of this paper is organized as follows. Section 2 describes details of the GBMA. Section 3 discusses the effects of applying the GBMA, referring to the simulation results. Section 4 describes related works. Finally, Section 5 presents a conclusion.

2. Geographically Bound Mobile Agent

The GBMA is a mobile agent whose location is restricted geographically. GBMA must be hosted on a node located in a designated region, until the tasks are completed. When the current host node of a GBMA leaves the designated region, the GBMA should migrate to another node in the designated region to remain in the region. Restriction of the GBMA location provides two advantages. First, communications between the GBMA and nodes in the region will be relative stable and involve low overhead. Since they communicate via shorter length routes, these communications are hardly prevented by decouplings of MANETs and involve a small amount of messages. Second, locating a GBMA is easy and involves low overhead. Sender needs to discover the GBMA only from nodes in the designated region.

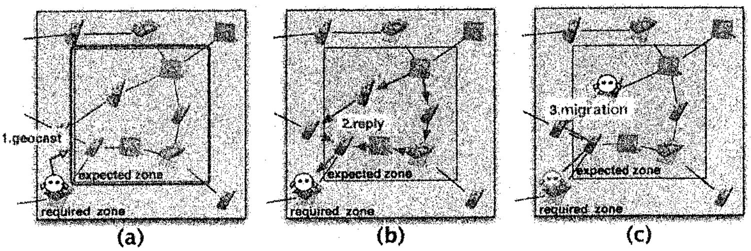


Figure 2. GBMA migration based on expected zone

However, it is difficult to determine when does start the remigration. The following actions are performed after starting GBMA migration until finishing migration: search for nodes in the designated region and routes to them, select a candidate node based on the aggregated node states, deactivate the GBMA, transfer its program code and data to the candidate node via the discovered route, and reactivate the GBMA on the candidate node. These actions cause migration latency. If a GBMA starts migration after its host node leaves the designated region, it remains outside the designated region until these actions are finished. While it remains outside the designated region, messages sent to it are not received. It must start its migration before its host node leaves the designated region. On the other hand, too early determination of the migration causes needlessly frequent migrations. To reduce the total duration for these migrations, the frequency of its migrations should be low. Therefore, the GBMA must start its migration at an appropriate timing. To define the designated region and to easily adjust the start timing of GBMA migration and the migration frequency, we propose two zones for the GBMA: the *required zone* and the *expected zone*.

2.1 Required zone and expected zone

A required zone is defined to clarify the region in which the GBMA should be located. The GBMA must be on a node in the required zone. The required zone, which restricts the GBMA location, can be also used when someone sends a message to the GBMA. The sender geocasts with a geocast region represented by the required zone. If the node receiving the message hosts the GBMA, the node passes the message to the GBMA.

On the other hand, an expected zone is defined to clarify the start timing and the frequency of GBMA migration. Figure 2 shows the GBMA actions based on the expected zone. The expected zone must be within the required zone. The GBMA continues to execute its own tasks while it is in the expected zone. When it detects that its host node is outside the expected zone, it starts to search for other nodes located in the expected zone and routes to them (Figure 2 (a)), selects one (Figure 2 (b)), and migrates to it through the discovered route (Figure 2 (c)). A start timing and a frequency of GBMA migration can be easily adjusted by modifying the expected zone size.

2.2 Application example

We take up data retrieval in a post disaster area described in Section 1, and describe the behavior of the GBMA. An observer surveys goods needed by evacuees in several shelters via a MANET. For each shelters located far from the observer, the observer uses GBMAs to retrieve data about necessary goods.

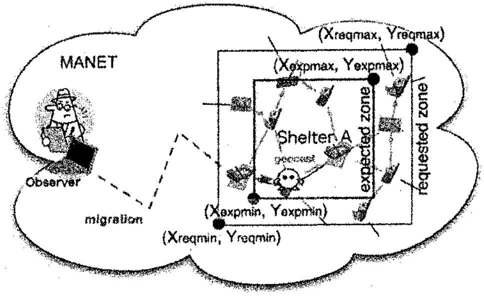


Figure 3. GBMA behavior example

Consider the case illustrated in Figure 3. The observer is interested in shelter A. First, the observer configures a required zone represented by coordinates (x_{reqmin}, y_{reqmin}) and (x_{reqmax}, y_{reqmax}) and an expected zone represented by coordinates (x_{expmin}, y_{expmin}) and (x_{expmax}, y_{expmax}) to the GBMA. (x_{reqmin}, y_{reqmin}) and (x_{reqmax}, y_{reqmax}) are set to a region that includes the shelter A. (x_{expmin}, y_{expmin}) and (x_{expmax}, y_{expmax}) are set to be within the requested zone: $x_{expmin} \geq x_{reqmin}$, $y_{expmin} \geq y_{reqmin}$, $x_{expmax} \leq x_{reqmax}$, and $y_{expmax} \leq y_{reqmax}$. Configuring (x_{expmin}, y_{expmin}) and (x_{expmax}, y_{expmax}), the start timing and frequency of GBMA migration can be easily adjusted.

Next, the GBMA on the observer node migrates to a node in the expected zone. GBMA migration protocol somewhat like Location-Aided Routing (LAR) [9] protocol. LAR is a source routing protocol based on DSR protocol[10]. Main difference between them is that a *route request* (RREQ) message in LAR protocol contains a destination node identifier, location of the destination node, and forwarding zone, and is delivered by directed flooding in the same way adopted in LBM protocol. The LAR RREQ can discover routes to a node which has same identifier contained in this RREQ and which is located in a region contained in this RREQ. After the route discovery, reply message is transferred to the observer node along the discovered route. In GBMA migration protocol, RREQ can discover routes to not one node but any nodes in the expected zone described by coordinates (x_{expmin}, y_{expmin}) and (x_{expmax}, y_{expmax}), and these nodes reply its route and its location. After that, the agent selects the node nearest to the coordinate $(\frac{x_{expmax}-x_{expmin}}{2}, \frac{y_{expmax}-y_{expmin}}{2})$ from among the received replies, and migrates to this node along the discovered route.

After migration, the GBMA in the requested zone starts to survey for necessary goods data in shelter A. The GBMA geocasts messages containing the required zone and the GBMA identifier, to the required zone. Evacuee nodes that receive this message send their own necessary goods data to the GBMA by LAR unicast, with the required zone and the GBMA identifier contained in the received message sent from the GBMA, each time necessary goods are added or modified. The notification messages are received by the GBMA as long as it is not migrating, at least one path from the notification source to it exists, and it is located in the required zone at the time. While aggregating the data, the GBMA periodically checks its host node location. If the GBMA is outside the expected zone, the GBMA starts to migrate in the same way as described above. The GBMA executes such migration and data retrieval.

The GBMA summarizes or filters aggregated data and compresses them in a manner befitting to this application, to reduce data size. Reduction of the data size reduces overhead of reporting back to the observer. On the other hand, if the GBMA deals with personal data, the GBMA needs to encrypt aggregated data. This application level encryptions prevents malicious host node from picking the data aggregated by the GBMA. If the observer wants to get the result, he or she sends a request to the required zone, and the GBMA receiving the request sends back the retrieved data via a route through which the request comes. Since the observer does not communicate with the GBMA until getting results, data retrieval does not prevent from decoupling of MANETs between the observer and the designated region, and he or she can moves freely.

Note that the expected zone size should be carefully chosen. Appropriately adjusting the size of the expected zone can reduce the duration for which the GBMA is outside the required zone. If the expected zone is too large, the GBMA leaves the required zone until migration is finished. Messages sent to the GBMA while the GBMA is outside the expected zone are lost. If the expected zone is too small, the GBMA frequently migrates and the duration for which the GBMA is migrating increases. The messages to the GBMA, which are sent while the GBMA is migrating are also lost. The GBMA with the optimal size of the expected zone may minimize the number of such lost messages. We examine this issue in Section 3.

3. Experiments and discussion

We evaluate the GBMA using the simulator implemented on JiST/SWANS [11]. Java in Simulation Time (JiST) is a high-performance discrete event simulation engine that runs over a standard Java virtual machine, and Scalable Wireless Ad hoc Network Simulator (SWANS) is a scalable ad hoc network simulator built atop the JiST platform. We implement the LBM protocol and the LAR unicast protocol, and develop the GBMA upon JiST/SWANS. With this simulator, we evaluate two issues concerning the GBMA: the reduction of the number of messages compared with the simple geocast approach with LBM described in Section 1, and the optimal expected zone size.

3.1 Simulation model

In the experiments, initially, nodes are distributed according to a grid over a rectangular region of 1000m × 1000m square described by the coordinates $(0, 0)$ and $(1000, 1000)$. Each node is equipped with an IEEE 802.11b wireless device and communicates with other nodes in the range of the wireless device. Each node is also equipped with a GPS device with which it can identify its location. Moreover, each node supports two mobility models: *static* and *random walk*[12]. In the static model, each node does not move, and in the random walk model, each node picks a direction randomly, walks a certain distance in that direction, pauses for some time, and repeats. Shelter A is rectangular in shape and defined by the coordinates $(600, 600)$ and $(800, 800)$.

3.2 Compared with Simple Geocast Approach

First, we compare the number of messages produced in a mobile agent approach, to that in the simple geocast approach. In this experiment, let the number of nodes be 12^2 and 14^2. Let the node mobility model be *static*. Furthermore, let the observer node be located at (x, x) where x is 600, 500, 400, 300, 200, and 100. In the simple geocast approach, the observer retrieves data from nodes in the shelter A. On the other hand, in the mobile agent approach, the observer sends a mobile agent to a node located in shelter A, and the mobile agent on the node retrieves the data. In both approaches, the observer (or the mobile agent) and the evacuees in the shelter A exchange messages 10 times. The simulation result is depicted in Figure 4.

In Figure 4, dotted lines represent results of the simple geocast approach, and solid lines represent results of the mobile agent approach. In the simple geocast approach, the number of messages increases exponentially, because the number of forwarding nodes increases according to the forwarding zone size. Therefore, the increment rate when the number of nodes is 14^2 is greater than that when the number of nodes is 12^2. On the other hand, in the mobile

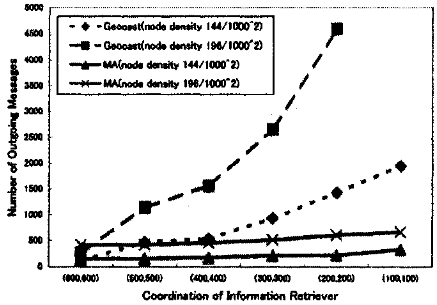


Figure 4. Total number of outgoing messages

agent approach, the increase in the number of the messages is smaller than that in the simple geocast approach. This is because, in the mobile agent approach, a mobile agent exchanges messages via shorter length routes, after migration to shelter A. Therefore, the distance affects the increase in the number of messages only when sending the mobile agent to shelter A, and does not affect message exchanges after migration. When x is 600, or the observer is located at shelter A, the number of messages in the mobile agent approach is slightly larger than that in the simple geocast approach. This is because, in this case, the data exchange overheads in both approaches are almost the same, but there is the overhead of mobile agent migration in the mobile agent approach. This result shows the mobile agent approach to be effective when the observer is far from the designated region, compared with the simple geocast approach.

3.3 Optimal expected zone size

Next, we compare the mobile agent and the GBMA, and examine how the expected zone affects the results. In this experiment, the number of nodes is 15^2 and the node mobility model is the *random walk* where the node walks $10m$ for 10 seconds and 20m for 10 seconds. Initially, the mobile agent or the GBMA is on the node at (700, 700); it is located at the center of shelter A. In the mobile agent approach, the mobile agent continues to retrieve data at the initial hosted node in spite of the node movement. In the GBMA approach, the GBMA migrates according to the node movement. The requested zone of GBMA is defined by the coordinates (600,600) and (800,800), and the expected zone is defined by the central coordinate of them and the length on the side being 200, 180, 160, 140, 120, 100, or 80 meters. When the length is 200 meters, the expected zone is the same region as the required zone. The smaller the side length is, the smaller the expected zone size is. The GBMA (or the mobile agent) initially geocasts a subscription message, and the nodes that receive it send messages to the GBMA at about one-minute intervals twenty

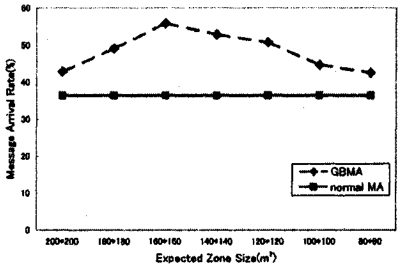


Figure 5. Message arrival rate(20m/10s)

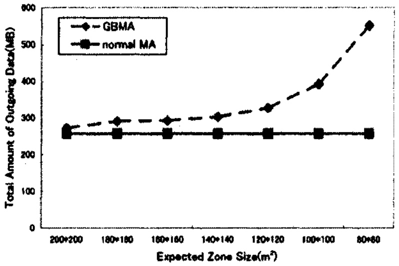


Figure 6. Total amount of outgoing data(20m/10s)

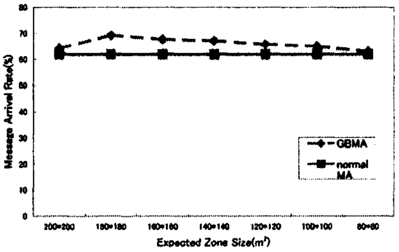


Figure 7. Message arrival rate(10m/10s)

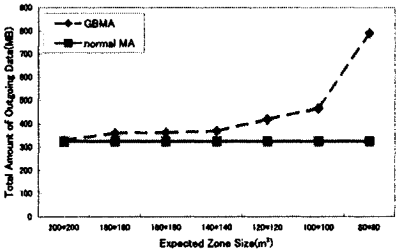


Figure 8. Total amount of outgoing data(10m/10s)

times. We measure the message arrival rate, which indicates the number of messages received by the GBMA (or the mobile agent) per total number of messages sent by the notification nodes, and the total amount of outgoing data. The simulation results are depicted in Figure 5, Figure 6, Figure 7, and Figure 8.

Figure 5 and Figure 6 show the message arrival rate and the total amount of outgoing data where the node speed is twenty meters per ten seconds, respectively. Similarly, Figure 7 and Figure 8 show where the node speed is ten meters per ten seconds, respectively. Figure 5 and Figure 7 show that the message arrival rate of the GBMA is always better than that of the mobile agent within the range of the expected zone sizes in this experiment. This is because the mobile agent not reacting to its host node movements goes outside the required zone where it does not receive the notification messages. On the other hand, the GBMA migrates to receive the messages in response to the node movement. The result shows that the GBMA effective for data retrieval, relative to the conventional mobile agent.

Let us consider the results of the GBMA. When the expected zone size is large, the message arrival rate is low. A large expected zone means that the GBMA starts to migrate relatively late in response to the node movement. Therefore, the duration for which the GBMA is outside the required zone becomes long. On the contrary, when the expected zone is small, the message arrival rate is also low. A small expected zone means that the GBMA frequently migrates. Therefore, the duration for which the GBMA is migrating becomes long. By appropriately configuring the expected zone size, the message arrival rate can be increased. In Figure 5, for an appropriate expected zone size, we observe that the rate improves by as much as 32%. Moreover, the results in Figure 5 and Figure 6 indicate that the optimal expected zone size will depend at least on the node speed. The optimal expected zone becomes bigger when the node speed becomes higher, because the time from when the node leaves the expected zone to when it leaves the required zone becomes shorter. The expected zone should be configured on the basis of the node speed. Figure 7 and Figure 8 show that the GBMA produces a larger number of messages than does the mobile agent. However, at the optimal expected zone size, the increment is not very large. In Figure 7, the increment with the optimal expected zone size is about only 11%. From these results, with the expected zone, the start time and frequency of GBMA remigration can be easily adjusted. Moreover, appropriate adjustment of expected zone size can improve the GBMA performance.

4. Related works

The battery problem is a major issue in a MANET[13]. Marinescu et al. proposed a method for conserving the battery of nodes by classifying the mobile devices on the basis of the node capability and specifying the role of each node in data transmission on the basis of that classification[14]. Song et al. proposed localized algorithms for energy-efficient routing structures[16]. Gitzenis and Bambos focused on data prefetch and proposed a method for minimizing the battery consumption of a certain node by postponing data prefetching when the link quality is low, and by proactively prefetching data items when the link quality is high[15]. Subramanian et al. proposed a battery-state-aware MAC protocol[17]. These strategies focus on message routing protocol or MAC protocol, and reduce the energy consumption in each message transmissions, but do not reduce the number of messages. Therefore, when many messages are transferred at application level, with only those methods, a large amount of battery power will be consumed. Applications using directed flooding with a location information[4][5][6][9] can reduce the number of messages by limiting node forwarding of the message, but does not provide sufficient reduction in some cases which we have described. We focus on application level, and propose a location-specific data retrieval application minimizing the number of messages, using a mobile agent.

Location tracking of the mobile agent is important for message exchanges with the mobile agent. Roth and Peters proposed a scalable and secure global tracking service for mobile agents with a method similar to the global hash table[18]. Li and Lam proposed a location update and search algorithm for tracking mobile agents[19]. Since these works target mobile agents in an internet, they will not be directly applicable to mobile agents in a MANET whose topology dynamically changes. We restrict the mobile agent location using a geographic data and realize simple tracking of the mobile agent. Moreover, by restricting the mobile agent location, efficient messaging to the mobile agent can be feasible by directed flooding with geographic data.

5. Conclusion

In this paper, we proposed the Geographic Bound Mobile Agent, the required zone, and the expected zone for a data retrieval of location-specific data in a MANET. With the GBMA migrating to remain near data sources, even when an observer is far from data sources and exchanges many messages with them, data can be retrieved with a small number of messages. Simulation results show that overhead of the GBMA approach is hardly affected by the distance between the observer and the designated region, whereas overhead of the simple geocast approach increases exponentially according to it. The required zone clarifies location of the GBMA and reduces overhead of looking up the GBMA, and the expected zone easily adjusts a start timing and a frequency of GBMA migration. Simulation results show that message arrival rate in the GBMA approach with appropriate expected zone size is 32% greater than that in the conventional mobile agent approach, in the case that each node speed is about $20m/10sec$.

Some issues still remain unresolved. In this work, the expected zone size was static. Because the speed of each node is different and is dynamically changed, the expected zone size should be dynamically adjusted in response to the speed of the current host node, and change its shape on the basis of the direction of the node movement. Moreover, the GBMA selected a candidate node for its migration destination on the basis of only current node locations, but this is insufficient. We should consider a more sophisticated selection with more properties such as node movement speed or node movement direction. We will take up these issues in a future work.

References

[1] J. Macker and I. Chakeres, IETF working group for mobile ad-hoc networks (MANET) charter, http://www.ietf.org/html.charters/manet-charter.html, 2002.

[2] M. Andreas, L. Thomas, R. Thomas, K. Thomas and K. Holger, Design Challenges for an Integrated Disaster Management Communication and Information System, Workshop on Disaster Recovery Networks (DIREN), 2002.

[3] J.C. Navas and T. Imielinski, Geocast-geographic addressing and routing, Proceedings of International Conference on Mobile Computing and Networking (MobiCom), ACM/IEEE, 1997.

[4] Y.B. Ko and N.H. Vaidya, Flooding-based Geocasting Protocols for Mobile Ad Hoc Networks, ACM/Baltzer Wireless Networks (WINET) journal, Vol. 7-6, pp. 471-480, 2002.

[5] I. Stojmenovic, A.P. Ruhil and D.K. Lobiyal, Voronoi Diagram and Convex Hull-Based Geocasting and Routing in Wireless Networks, Proceedings of IEEE International Symposium on Computers and Communications, pp. 51, 2003.

[6] W.H. Liao, Y.C. Tseng, K.L. Lo and J.P. Sheu, GeoGRID: A Geocasting Protocol for Mobile Ad Hoc Networks Based on GRID, Journal of Internet Technology, Vol. 1-2, pp. 23-32, 2000.

[7] J. White, Mobile Agents White Paper, General Magic, 1996.

[8] R.S. Gray, et al., Mobile-Agent versus Client/Server Performance: Scalability in an Information-Retrieval Task, Proceedings of International Conference of Mobile Agents (MA), 2001.

[9] Y.B. Ko and N.H. Vaidya, Location-Aided Routing(LAR) in Mobile Ad hoc Networks,

[10] D.B. Johnson, D.A. Maltz, and J. Broch, DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad-Hoc Networks, Ad Hoc Networking, edited by Charles E. Perkins, Chapter 5, pp. 139-172, Addison-Wesley, 2001.ACM/Baltzer Wireless Networks (WINET) journal, Vol. 6-4, pp. 307-321, 2000.

[11] R. Barr, Z.J. Haas and R. Renesse. JiST:Embedding Simulation Time into a Virtual Machine, Proceedings of EuroSim Congress on Modelling and Simulation (EuroSim), 2004.

[12] T. Camp, J. Boleng and V. Davies, A Survey of Mobility Models for Ad Hoc Network Research, Wireless Communication and Mobile Computing (WCMC): Special Issue on Mobile Ad Hoc Networking Research, Trends and Applications, Vol. 2, No. 5, pp. 483-502, 2002.

[13] C.E. Jones, K.M. Sivalingam, P. Agrawal and J.C. Chen, A Survey of Energy Efficient Network Protocols for Wireless Networks, Wireless Networks, Vol. 7, No. 4, pp. 343-358, 2001.

[14] D.C. Marinescu, G.M. Marinesxu, Y. Ji, L. Boloni and H.J. Siegel, Ad Hoc Grids: Communication and Computing in a Power Constrained Environment, Workshop on Energy-Efficient Wireless Communications and Networks (EWCN), 2003,

[15] S. Gitzenis and N. Bambos, Efficient Data Prefetching for Power-Controlled Wireless Packet Networks, Internetional Conference on Mobile and Ubiquitous Systems (MobiQuitous), 2004.

[16] W.Z. Song, Y. Wang and X.Y. Li, Localized Algorithms for Energy Efficient Topology in Wireless Ad Hoc Networks, International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), 2004.

[17] J. Subramanian, M. Bs and S.R. Murthy, On Using Battery State for Medium Access Control in Ad hoc Wireless Networks, International Conference on Mobile Computing and Networking (MobiCom), 2004.

[18] V. Roth and J. Peters, A Scalable and Secure Global Tracking Service for Mobile Agents, Proceedings of International Conference of Mobile Agents (MA), 2001.

[19] T.Y. Li and K.Y. Lam, An Optimal Location Update and Searching Algorithm for Tracking Mobile Agent, Proceedings of International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2002.

OPTIMIZATION OF HANDOVER PERFORMANCE FOR FMIPV6

Li Jun Zhang[1], Samuel Pierre[1] and Laurent Marchand[2]

[1]Mobile Computing Networking Research Laboratory (LARIM), Department of Computer Engineering, École Polytechnique de Montréal, C.P. 6079, succ. Centre-Ville, Montreal, Que., Canada H3C 3A7; [2]Ericsson Research Canada, 8400, Decarie Blvd, Town of Mount Royal, Que., Canada H4P 2N2

Abstract: This paper presents a new protocol, namely Access Routers Tunneling Protocol (ARTP), dedicated to pre-configuring bidirectional secure tunnels among adjacent access routers before handoff. This protocol allows two tunnel endpoints to negotiate quality of service-related parameters, traffic classification aspects, security policies, such as authentication and encryption methods, buffering mechanism, etc. Once the parameters of pre-established tunnels are determined, real-time traffic could be redirected in a cost-efficient way to mobile users using GRE (Generic Routing Encapsulation) tunneling technique. This protocol allows us to optimize handover performance for FMIPv6. An existing analytical model is used to evaluate the performance of the proposed handover procedure. Numerical results show that our new approach has better performance than FMIPv6 in terms of signaling cost, and the buffer size required during handoff.

Key words: fast handover; bidirectional secure tunnels; handover latency; performance analysis.

1. INTRODUCTION

User mobility and real-time data traffic (e.g. Voice over IP) are two expanding areas within communication systems. On one hand, in order to guarantee user mobility, handover has to be taken into account in mobile networks, where subscribers move around. On the other hand, transporting real-time traffic to the IP-enabled mobile user imposes strict requirements on latency and packet loss. As mobile users roam in the network, they frequently change their point of attachment to the network. Therefore it is necessary to keep the continuity of communication in progress, and the access network should provide features of minimizing the interruption to ongoing sessions. However, controlling the handover mechanism is quite complicated in mobile networks.

Based on these contexts, we propose a new protocol with the purpose of minimizing handover latency, packet losses and jitter for real-time service. This protocol describes mechanism of pre-configuration of bidirectional secure tunnels among adjacent access routers. With the pre-established tunnels, a mobile node can resume its previous ongoing session immediately after performing L2 handoff at the visited network; moreover, it can initiate a real-time session using its previous care-of-address upon arrival on the new link. By this means, access routers are equipped with the flexibility of offering service with guaranteed quality to their neighbors' subscribers.

The rest of this paper is organized as follows. Section 2 describes the principles of the handover procedures found in recent literature. Section 3 proposes the Access Routers Tunneling Protocol, and the proposed handover procedure for improving handover performance of FMIPv6. Section 4 presents an analytical model to evaluate the performance of our new approach; numerical results are also illustrated and compared with FMIPv6.

2. BACKGROUND AND RELATED WORK

Recently, fast and seamless handover procedures for IP-based communication networks have become hot topics in the field of mobility management. Since in the future mobile communication networks, a user is able to conveniently roam between various operators and between fixed and mobile as well as public and private networks independently of the different access technologies used, improving handover performance is quite significant. Furthermore, it is essential to support real-time applications which deal with tight time constraints for offering adequate quality of service and to deploy all-IP networks which are cost efficient comparing with the current network infrastructure in the next generation wireless networks. However, synchronous real-time applications such as Voice over IP and Video Conference over IP place new demands on the quality of IP services: packet loss, delay variation or jitter need careful simultaneous control; these requirements impose strong challenges in mobile environments.

2.1 Fast Handover for Mobile IPv6

IETF proposed the approach called Fast Handover for Mobile IPv6 (FMIPv6) with the intention of minimizing the handover latency in MIPv6. FMIPv6 allows a mobile user to pre-configure a new on-link care-of-address before breaking its connection with the previous access router (PAR) [1].

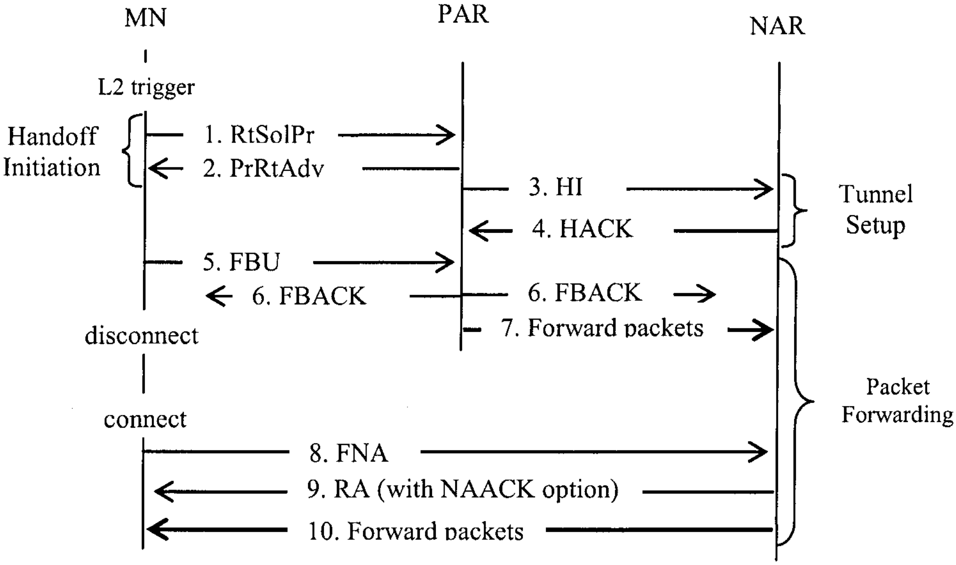


Figure 1. Fast handover with anticipation in FMIPv6

Fast handover is triggered when a mobile node (MN) receives L2 trigger before it moves to the new network. This mobile then sends a *Router Solicitation for Proxy Advertisement* (RtSolPr) message to the PAR asking for resolving the Access Point Identifiers to subnet-specific information. The PAR replies with a *Proxy Router Advertisement* (PrRtAdv) message to the MN. Based on this message, the mobile node generates a new on-link care-of-address, and then sends a *Fast Binding Update* (FBU) to the PAR, including its prospective care-of-address on the new link. During the movement of MN, the PAR sends a *Handover Initiate* (HI) message to the new access router (NAR) to initiate the tunnel setup process. After verifying the uniqueness of the MN's new care-of-address, NAR sends back a *Handover Acknowledgment* (HACK) message to PAR as a reply to the HI message, thus a temporary bidirectional tunnels are established between the two access routers. Consequently, the PAR sends *Fast Binding Acknowledge* (FBACK) to the MN. Once the PAR intercepts packets destined to the mobile node, it tunnels the packets to the NAR. Upon arrival at the new subnet, the MN sends a *Fast Neighbor Advertisement* (FNA) to the NAR to announce its attachment and also to confirm the validity of new on-link care-of-address in case where MN has not received the FBACK on the previous link. Upon receipt of the FNA, the NAR delivers the packets to the MN. Figure 1 shows the fast handover procedure with anticipation in FMIPv6.

2.2 Buffer Management Scheme for Fast Handover

When a mobile user roams from one network to another, there is always an inevitable link down time during handoff which leads to packet loss. This would have bad effect on the quality of communication. To avoid packet drops, a feasible solution is to buffer those in flight packets sent by correspondent nodes. However, the original fast handover protocol, namely FMIPv6, does not support buffering mechanism during a pure link layer handoff [2]. This means that an access router is unable to buffer packets for a mobile user when it is moving between different access points (base stations) within the same subnet, thus the temporary disconnection is unavoidable and results in packet loss. Under this circumstance, an enhanced buffer management scheme is proposed to improve buffer utilization on access routers as well as to support QoS services during handover process [2].

The principal ideas are: buffering implemented both in PAR and in NAR, and three types of services, namely real-time traffic, high priority and best effort traffic, are defined so that packets can be treated differently based on their traffic characteristics. Handover procedure is triggered by specific link layer events or policy [2]. Upon receipt of this trigger, the mobile node sends a request of B*uffer Iinitiation* (BI) message piggybacked in the *Router Solicitation for Proxy Advertisement* (RtSolPr) to the PAR for requesting the buffer space. While the establishment of a bidirectional tunnel between PAR and NAR, the allocation of buffer space for the MN is also negotiated via the *Buffer Request* (BR) and *Buffer Acknowledge* (BA) messages. Subsequently, the PAR sends a *Proxy Router Advertisement* (PrRtAdv) message to the MN indicating the success of allocation of buffer space, and informing it of the new subnet prefix. With this message, MN generates a new on-link care-of-address (NLCoA) and includes this address in a *Fast Binding Update* (FBU) sent to PAR. Upon receipt of the FBU, the PAR starts buffering packets and/or forwards them to NAR. While connecting to NAR, the mobile node sends a *Buffer Forward* (BF) message to both the PAR (via the NAR) and the NAR. Thereafter, the two access routers forward packets in their buffers to the MN. Figure 2 shows the handover procedure with buffer scheme.

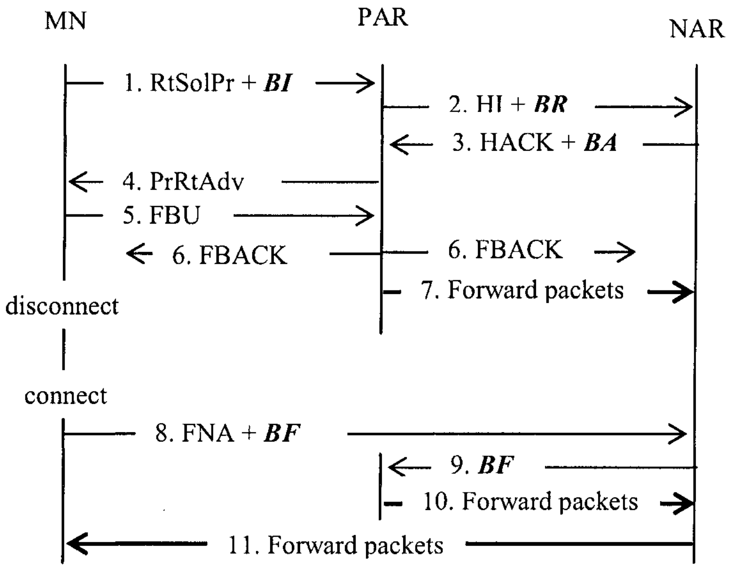


Figure 2. Handover procedure with buffering scheme

Recent work has been directed at improving handover performance to support real-time traffic. However, in order to provide successful real-time services, it is necessary to minimize the traffic redirection in mobile environments. Whether a mobile user has the right to obtain specific routing treatment depends on whether it negotiated a successful Authentication, Authorization and Accounting (AAA) exchange with a network access server at some point of the past [3, 4]. Furthermore, the mobile node for which the context transfer protocol operations are undertaken is always identified by its previous care-of-address [4]. Therefore, we propose a new protocol dedicated to pre-establishing bidirectional secure tunnel before actual handoff so that mobile nodes could use their previous care-of-address in a visited network. By this means, packet losses and handover latency could be reduced. Furthermore, since the pre-configured tunnels support quality-of-service (QoS) by traffic classification mechanism, local resource reservation as well as admission control, the disruption for real-time ongoing session can be minimized significantly.

3. ACCESS ROUTERS TUNNELING PROTOCOL

The Access Routers Tunneling Protocol (ARTP) is a new signaling protocol to setup tunnel parameters between two access routers. ICMP-type messages are defined and used to carry information of QoS-related parameters, authentication method, encryption method, service class, etc. so as to facilitate the negotiation between two tunnel endpoints. Concerning the security aspects, two mechanisms are deployed to secure the traffic: session key generated by access router and tunnel token formulated by mobile node.

3.1 Tunnels Setup Algorithm

The algorithm for setup the tunnels is described as follows:

```
1)  request = 0;    request_MAX = 4;            neighbor_indice=0;
2)  Tunnel brokers at access routers create their neighbor tables.
3)  AR_1 selects one entry from its Neighbor Table.
4)  /*verify the reachability of the neighbor*/
    if(the selected neighbor: AR_2 is reachable) {
5)        request=request+1;
6)        if(request < request_MAX) {
7)                AR_1 sends a tunnel Request message to AR_2;
                  AR_2 verifies its capability;
                  AR_2 proposes parameters with Tunnel Reply message;
                  AR_2 sends this Tunnel Reply to AR_1;
8)                if(AR_1 accepts the condition) {
                        AR_1 sends a Tunnel_ACK to AR_2;
9)                      if(tunnels is symmetric) /*symmetric tunnel*/  {
                            with the negotiation results,
                            AR_1&AR_2 add an entry in Forward Tunnel table;
                            AR_1&AR_2 add en entry in Reverse Tunnel table;
                            go to END; }
10)                     else /* in case of asymmetric tunnel*/ {
                             AR_1 adds an entry in its Forward Tunnel table;
                             AR_2 adds an entry in its Reverse Tunnel table;
                             /* reverse tunnel setup procedure*/
                             AR_1 sends AR_2 a Reverse Tunnel Request message;
                             AR_2 sends a Tunnel Request to AR_1;
                             AR1 responses with a Tunnel Reply;
11)                          if(AR_2 accepts the proposed parameters of AR_1) {
                               negotiation = true;
                               AR_2 adds an entry to its Forward Tunnel table;
                               AR_1 adds an entry to its Reverse Tunnel table;
                               go to END; }
                             else { negotiation = false; go to END;  }
                        }/* end of reverse tunnel*/
                    }
                    else /* another negotiation*/ { go to step 5;  }
              }
              else /*request > request_MAX*/ { go to END; }
        }
        else /* in case neighbor is unreachable*/  {
              go to step 3 ;
              neighbor_indice ++ ;   /* select another neighbor*/ }
    END;
```

3.2 The Proposed Handoff Scheme

The proposed handover algorithm allows a mobile node to resume its real-time ongoing session with its correspondent as soon as it attaches to the new link. With the preconfigured bidirectional tunnels, traffic will be redirected to the new network using the MN's previous care-of-address. By this means, the service disruption for an on-going real-time session could be minimized. Figure 3 shows the overall handover procedure.

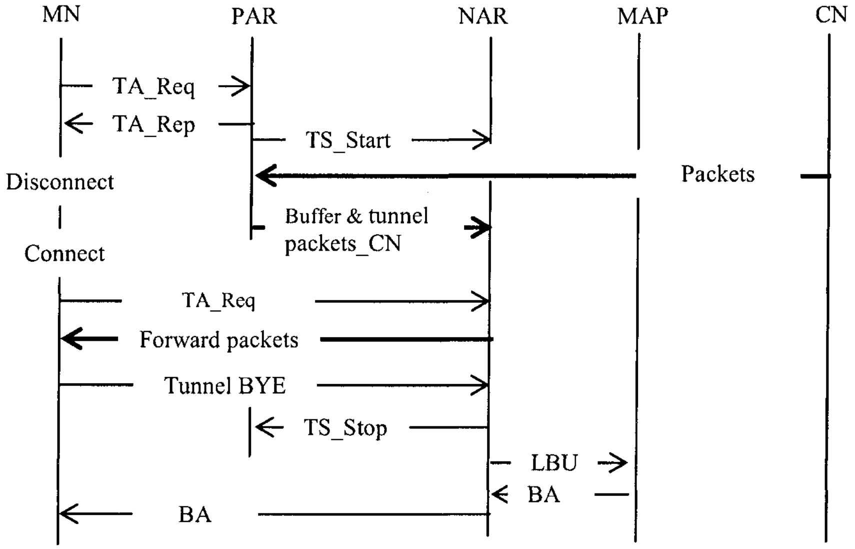


Figure 3. Proposed handover procedure

Before actual handover, adjacent access routers have established business relationships so that bidirectional secure tunnels have already been created. Handover is triggered by specific link layer event. A mobile user roams with a real-time session in course. Before the MN breaks the connection with the PAR, it sends a Tunnel Activate Request (TA_Req) message to the PAR. Upon receipt of this message, the PAR performs local resource reservation for the mobile and sends a Tunnel Activate Reply (TA_Rep) to the MN; meanwhile, it sends a Tunnel Session Start indication to the new access router (NAR) with the bearer context of the mobile. When the correspondent node (CN) sends packets to the MN, the PAR intercepts the packets, buffers them and tunnels to the NAR. Upon receipt of the TS_Start indication, the NAR performs admission control and also reserve the required bandwidth for the imminent MN. As the mobile arrives on the new link, after the L2 handover, it may initiate a new real-time session or just send a TA_Req to the NAR using its previous care-of-address. The NAR then forwards packets to the MN. Once the session in course is complete, the MN sends a Tunnel BYE message to the NAR to deactivate the tunnel. The NAR releases the reserved resource and sends a Tunnel Session Stop message to the PAR requesting the PAR to deactivate the session; meanwhile, the NAR assigns a new care-of-address to the MN and sends a local binding update (LBU) to the MAP on behalf of the MN. Accordingly, the MAP modifies its binding cache, and reply with a *Binding Acknowledgement* (BA) to the NAR which then forwards the BA to the MN.

4. PERFORMANCE ANALYSIS

We use an existing analytical model and the reference values found in the literature [5] to evaluate the performance of our new approach. Table 1 and Table 2 illustrate the parameters used to get numerical results. With the same principle as the analytical model [5], we obtain Figure 4 and Figure 5.

Table 1. System parameters for signaling cost

α	β	γ	$C_{transmit\_MP}$	$C_{transmit\_PN}$	$C_{process\_PAR}$	$C_{process\_NAR}$	C_{signal_MIPv6}
0.2	0.8	1.0	10	2	5	5	100

Table 2. System parameters for packet delivery cost

δ	ε	λ	$t_L + t_I$	t_R	t_{PAR_NAR}	t_{BU}	t_{New}	t_{CN_PAR}	t_{MN_NAR}
0.2	0.8	1.0	165	10	5	160	160	150	10

Figure 4 shows the signaling cost comparison as L2 trigger time changes in case where the decreasing factor equals to 0.5. As shown in Figure 4, the ARTP-based handover scheme has better performance than FMIPv6 in terms of signaling cost because bidirectional secure tunnels are established before actual handoff. The average signaling cost of ARTP-based handover is 118.9, compared to 129.1 for FMIPv6, the gain is 7.90%; compared to 129.4 for buffer-based Handover, the average gain is 8.11%. As L2 trigger time elapses, the signaling cost of FMIPv6 and buffer-based HO converges to certain value. However, FMIPv6, buffer-based HO and ARTP-based HO have more important signaling cost than MIPv6 because ARTP-based HO aims to improve the performance of FMIPv6 without intention to minimize the signaling overhead of MIPv6.

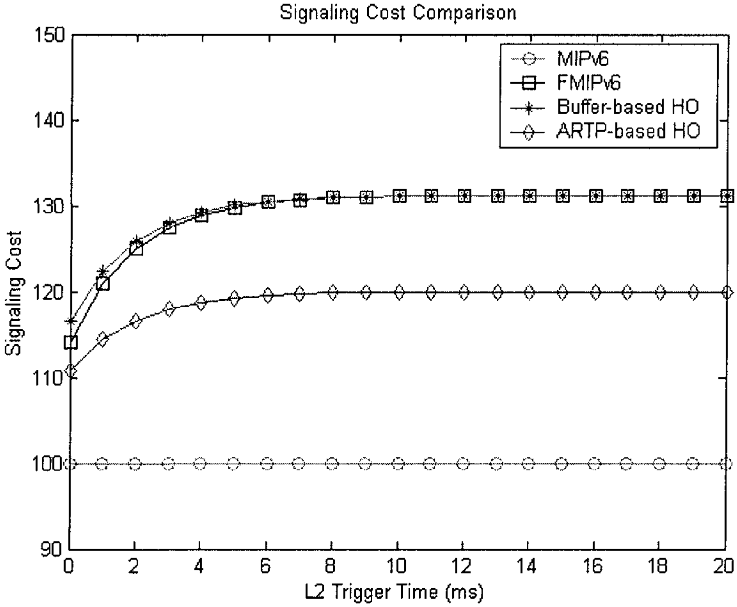


Figure 4. Signaling cost comparison as L2 trigger time changes

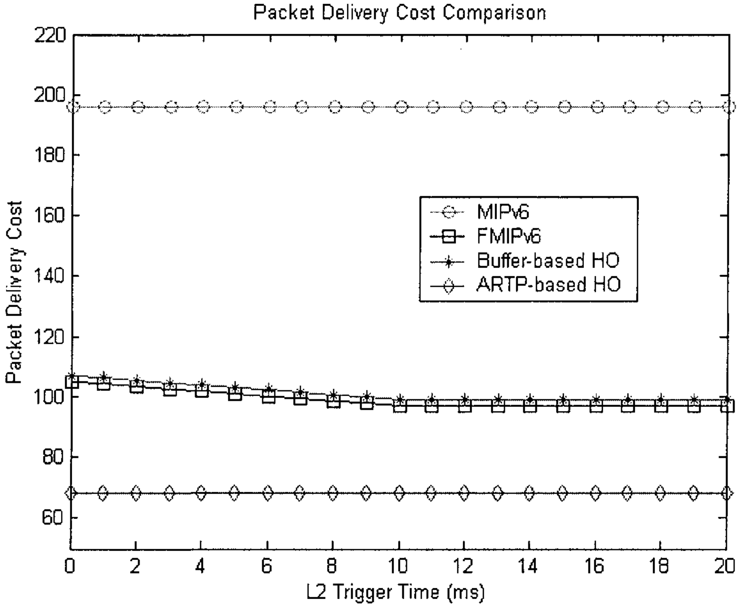


Figure 5. Packet delivery cost comparison as L2 trigger time changes

From Figure 5, we find that ARTP-based handover has better performance than FMIPv6 in terms of number of buffered packets during handoff. The average packet delivery cost of ARTP-based handover is 68.0, compared to 99.1 for FMIPv6, the gain is 31.38%; compared to 101.1 for buffer-based Handover, the average gain is 32.74%; compared to 196.0 for MIPv6, the average gain is 65.31%. As shown in Figure 5, L2 trigger time has less influence on packet delivery cost. Since the cost is defined as the number of packets buffered during handoff, it is proportional to the packet arrival rate and the handover latency. In our example, the handover latency in MIPv6 is more important than FMIPv6, more buffer space is required in MIPv6. In addition, we can find that the handover latency in the ARTP-based HO scheme is much shorter than in FMIPv6.

5. CONCLUSION

In this paper, we proposed a new protocol for pre-establishing bidirectional secure tunnels among adjacent access routers. Using the preconfigured tunnels, handover latency and the required buffer during handoff can be reduced significantly. Numerical results show that the ARTP-based handover scheme has better performance than pure FMIPv6 and the buffering-based handover scheme in terms of signaling cost and the number of buffered packets during handover. In addition, service disruption for real-time ongoing session could also be minimized. This protocol also allows access routers to provide certain quality of service to their neighbors' clients as the QoS-related parameters are negotiated on the basis of service class prior to handoff process. Further performance comparison will be done with realistic workloads through implementation and simulation.

REFERENCES

1. R. Koodli, "Fast Handovers for Mobile IPv6", draft-ietf-mipshop-fast-mipv6-03.txt, October 2004.
2. W.M. Yao, Y.C. Chen, "An enhanced buffer management scheme for fast handover protocol", the 24th International Conference on Distributed Computing Systems Workshops Proceedings, Mar. 2004, pp. 896 – 902.
3. J. Kempf, "Problem Description: Reasons for Performing Context Transfers Between Nodes in an IP Access Network", RFC-3374, September 2002.
4. J. Loughney, M. Nakhjiri, C. Perkins, "Context Transfer Protocol", draft-ietf-seamoby-ctp-11.txt, August 2004.
5. S. Pack, Y. Choi, "Performance Analysis of Fast Handover in Mobile IPv6 Networks", IFIP PWC 2003, Venice, Italy, September 2003.

XML ACCESS CONTROL FOR SECURITY AND MEMORY MANAGEMENT

Sun-Moon Jo[1], Chang-Mo Yang[2], Weon-Hee Yoo[1]

[1] Department of Computer Science and Information Engineering,
Inha University
253 YongHyun Dong, Nam Ku, Incheon, Korea
sunmoonpink@hanmail.net, whyoo@inha.ac.kr
Department of Computer Education
[2] Sugok-dong, Heung duk-gu, cheongju, Chungbuk, Korea
cmyang@cje.ac.kr

Abstract. Since XML was presented as a standard data type on the web, many data have been made and transformed into the XML type, consequently generating a large amount of XML data. Therefore, the need for efficient management and security of large-capacity XML data is gradually becoming important. The existing access control has problems that DOM trees should be loaded on memory in the process of parsing all XML documents to generate DOM trees, that a large amount of memory is used to search for trees repetitively to set access authorization on all nodes of DOM trees, and that the system becomes inefficient due to complicated authorization assessment. In this paper, we suggest an access control policy model and tree labeling algorithm for secure XML documents. So it can reduce expenses of authorization assessment of the existing access control implemented in a complicated and repetitive way.

1 Introduction

After XML (eXtensible Markup Language) was presented as a standard for data exchange and representation on Internet, many new data have been made in the XML type and the existing data have been transformed into the XML type; consequently, the amount of XML data is increasing drastically [10]. XML can use its merit of describing meaningful information immediately to provide a standard data type in the form of exchanging information on a lot of data generated in the process of companies' database or applied program operation. It is therefore very appropriate for a component label and document management system that needs definition and description of detailed information and its meaning. As a large amount of XML-type information was provided on web environment, developers and users became more concerned about the issue of XML document security.

* This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD)" (R05-2004-000-11694-0)

As for researches in XML document security and relevant products, control of access to information in web environment and transmission layer security protocols including electronic signature and coding is mostly related to HTML documents, and couldn't deal with access control according to the meaning of partial information, which is the main advantage of XML as file-based access control. So access control applying the advantage of XML became necessary [5]. The existing access control first parses XML documents to get DOM trees if a user demands XML documents. After the parsing, it sets the sign value that means permission (+) or denial (-) of access to nodes of DOM trees in reference to authorization of relevant database and XML documents. Nodes with the sign value set at - in DOM trees are removed and only those with the value set at + are shown to the user in the XML type [3], [4], [5].

However, it has problems that DOM trees should be loaded on memory, that a large amount of memory is used to search for trees repetitively to set access authorization on all nodes of DOM trees, and that the system becomes inefficient due to complicated authorization assessment.

In this paper, we suggest an access control policy model and tree labeling algorithm for secure XML documents. It is therefore possible to make it easy to manage information on access authorization and users and remove unnecessary parsing and DOM tree searching, consequently obtaining better efficiency than the existing access control model.

The paper is organized as follows. Section 2 examines studies and problems about XML access control. Section 3 defines the concept of XACML and an action label type group(ALTG) for access control policy models to describe tree labeling algorithm. Section 4 evaluates the access control policy models and section 5 draws a conclusion and describes the future course of studies.

2 DOM-Based XML Access Control

An XML DOM tree provides API (Application Program Interface) to access elements of XML documents [2]. The existing access control models [1], [3], [4] uses such a DOM tree to set access authorization to elements of DTD and XML documents and control users' access to XML data according to information on access authorization set.

According to the process of changes in documents in Figure 1, there is a request for seeing XML documents. As for all XML documents and DTD concerned, information on access authorization is specified in documents called XAS (XML Access Sheet). XML documents are parsed to obtain DOM trees; then, a value of sign is set which means admission (+) or rejection (-) of access to nodes of DOM trees based on XAS of DTD and XML documents. It is called labeling to set authorization to nodes of DOM trees. The nodes with the value of sign set as - are removed from the labeled DOM trees and only those with the value set as + are restored to the user [1], [3], [4], [5]. Here, although XML documents with nodes removed from DOM trees can fail to be valid (its solution requires the loosening process, with all elements and attributes set as optional in DTD), they can maintain the existing DTD despite the removal of nodes from DOM trees.

To solve the problem that XML documents with nodes removed from DOM trees can fail to be valid for DTD, the loosening technique is suggested to maintain the existing DTD despite the removal of nodes from DOM trees [3]. However, this method causes a semantic conflict due to the loss of information on the structure. The tree labeling technique is used to maintain information on the structure of documents, which has a problem that it can violate secrecy by showing the existence of data and information on the structure with rejection (-) labeling [3], [4]. Although it applies a strong labeling technique to prohibit access to nodes with rejection (-) labeling [1], this technique has limitations in usability of data by prohibiting access to sub-nodes of prohibited nodes.

Above-mentioned studies have a problem of reducing the efficiency of system as the entire DOM trees should be loaded in memory and much memory is used due to repetitive tree retrieval to set access authorization to all nodes in DOM trees.

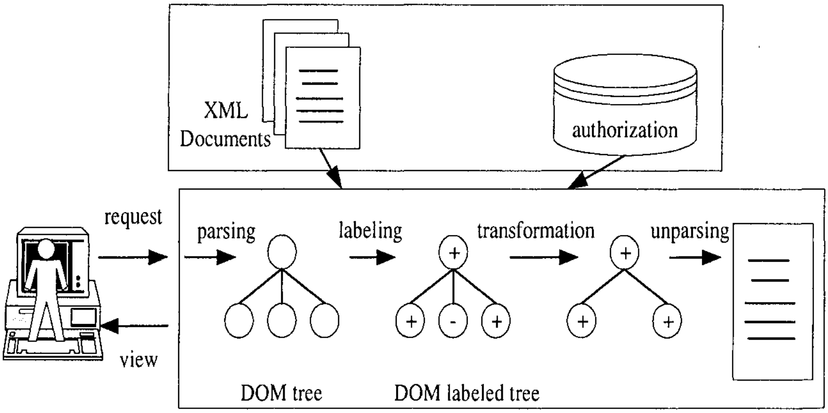


Fig. 1. XML Document Access Control Processor

3 Access Control Model for XML Documents

In this section, we discuss access control policies and tree labeling algorithm technique for securing XML documents.

3.1 XACML Concept

XACML(eXtensible Access Control Markup Language) is an international standard on access control [6], which is composed of policy language described by XML and access control decision request and response language. XML-based access control is composed of XML vocabularies to express rules on authorization. It provides minute access control services for resources requiring security by using XML vocabularies to define access control rules. An access control policy is to determine who can carry out which operation on which resource.

Policy language, which describes general requirements for access control on rules, policies, and policy set, also defines functions, data types, combining logic, and so on. Request language serves to construct questions about which object can perform a specific action for a particular resource; response language is used to express results of the request, with responses indicated in four results: permit, deny, indeterminate, and not applicable [8], [9].

3.2 Requirements for XML Access Control System

The existing web-based access control models can describe authorization in a unit or part of files. However, this method fails to make access based on a meaning of information in order to deal with information by the meaning, which characterizes XML documents most remarkably, or access to such small units as elements. Therefore, requirements for access control to XML documents can be summarized as follows [5]:

① Authorization should be provided in many structural levels.

② Extension to existing Web server technology. XML documents are usually made available by means of Web sites, using a variety of HTTP-based protocols.

③ It is necessary to support fined-grained access control. The access control model should provide access control in many levels such as document set or one element.

④ It is necessary to secure transparency. If it should be transparent to a user to conduct operations of an access control system, it should be impossible to know which part is provided with no authority in a document a requester looks at.

⑤ Smoothness integration with existing technologies for user authentication (e. g. digital signatures). Access control should complement tag level authentication based on digital signatures.

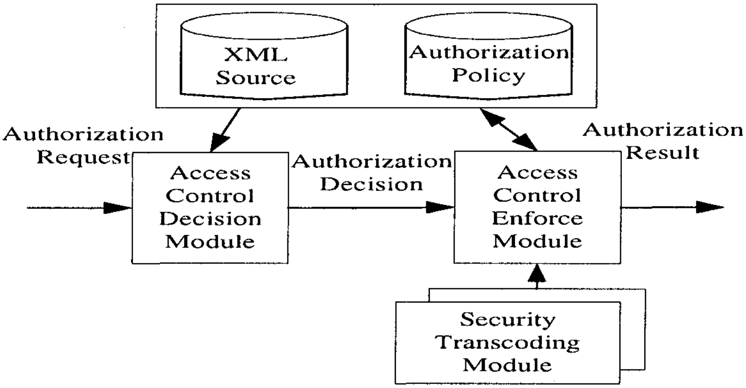


Fig. 2. Access Control Architecture for XML Documents

The Figure 2 shows the structure of an access control model for XML documents. A user requests access to resources in a system; the system determines whether to admit or reject by referring to information on access control policy and XML documents requested by ACM (Access Control Module) after confirming that the requester is a legitimate user. This determination is transmitted to ACM (Access Control Module); in case of a request for operation and reading of XML documents concerned, the documents are manufactured only with information which a user is authorized to read and then transmitted to the user [7].

3.3 Authorization Subjects and Authorization Objects

In our model, a subject is a user. It is not the purpose of this paper to give detailed information on how these subjects are organized. Each user has an identifier which can be the login name. Each user is the member of one or several user groups. For the sake of simplicity we assume that groups are disjoint but they can be nested.

An object is Resource pointer of target object such as a directory path expression. Since we deal with XML documents in this paper, we use an XPath [11] tree to specify the target objects.

3.4 Access Authorizations Policy Rules

XML document access authorizations are composed of subject, object, action, action label type group, sign, and type:

- Subject: User name, IP address, computer name (A subject who accesses XML documents and provides user group and pattern);
- Object: XPath 1.0 (An element of XML documents, which is expressed in XPath);
- Action: Read, write, create, delete (An operation the subject can implement);
- Action Label Type Group: R(read operator group), DSLG(Document and Structure Label Group; operator group);

- Sign: {+, -} is the sign of the authorization, which can be positive (allow access) or negative (forbid access);
- Type: {L, R, LDH, RDH, LS, RS, LD, RD} is the type of the authorization (Local, Recursive, Local DTD Hard, Recursive DTD Hard, Local Soft, Recursive Soft, Local DTD, and Recursive DTD, respectively).

The subject of authorization can be described as id or the access-requested position. The object means a resource to protect access. XPath language, which is a W3C standard of path expression, or expanded Uniform Resource Identifier (URI), is used to express the object [11]. Path expression is a list of pre-defined functions or element names differentiated by a divider (/) on the structure of document tree. Action refers to an operation the subject can implement; according to how much action label type group(ALTG) operators change XML, the operator group can be classified as follows:

- Read Label Group: A set of operators that read but never change documents in XML (Read).
- Document Structure Label Group: A set of operators that change XML documents and structures (Insert, Delete, Rename).

If a new operator is added to an access control model, information on access authorization becomes complicated because the operator's information should also be included in the information on access authorization. And labeling and DTD verification processes reduce the efficiency of the system due to repetitive DOM tree retrieval and parsing.

3.5 Propagation Policy Rule

A Propagation policy rule is a security policy to use for regulating authorization conflicts to set access authorization. As for authorization interpretation, the final sign (+ or -) is determined by reflecting propagation and overriding in each element. If there are both permission and denial for the same subject, only one access authorization is determined according to the conflict settlement principle. The following steps are rules to determine precedence of authorization in case of authorization conflicts.

Rule 1: Authorization on the most specific subject described according to partial order of subjects takes precedence.

Rule 2: Directly described authorization rather than that occurring by transmission takes precedence.

Rule 3: Authorization directly described on XML documents rather than that described on DTD takes precedence.

Rule 4: Authorization on nodes rather than that of its forefather takes precedence.

3.6 Default Policy

When there is no permission(grant or deny) for an access request or when the confliction resolution policy “nothing takes precedence” is enforced, we need to make a decision according to the specified default policy. This can be specified in the <default> element for each action The default policy is “deny” by default for every action.

3.7 Access Control Technique

Labeling is the process of using information on access authorization defined by a security manager to set access authorization to nodes of DOM trees requested by a user. The information on labeled authorization is used in determining whether to admit or reject the user's request. To label information on authorization to DOM trees based on an operator, it was necessary to repeat the labeling process as many times as the number of kinds of operators included in a question. Suggested is labeling algorithm based on the ALTG to remove such a repetitive labeling process.

■ Document Tree Labeling Algorithm ■

```
Input : A requester rq and an XML document URI

Output : The view of the requester rq on the document URI

Method : /* L is local, R is recursive, LDH is Local DTD Hard, RDH is Recursive DTD Hard, LS is Local Soft, RS is Recursive Soft, LD is local DTD-level, RD is recursive DTD-level */

1. A.xml A = {a= <subject, object, action, ALTG, sign, type> | a ∈ authorization, rq ≤ AS subject, uri(object)= =URI OR uri(object) ==(URI)}

2. Let r be the root of the tree T corresponding to the document URI, n is a node other than r, p is the parent node of n

3. AM( ) : returns the ALTG of a node specified in the authorization rule,

4. Type() : returns the type specified in the authorization rule,

5. Propagation_rule() : returns the ALTG determined by propagation rules

6. For each c ∈ children(r) do label(c, r)

7. For each c ∈ children(r) do prune(T, c)

8. L1r = AM(r) in A.dtd ,  L2r = AM(r) in A.xml

9. initial_label(r)

10. For each c ∈ children(r) do label(n,p)
```

```
Procedure initial_label(n)

     if  L1r ∪ L2r ={ }, Lr=default(r)

     else  Lr = propagation_rule([L1r, L2r])

Procedure label(n,p)

if type (p) in [L, R, LDH, RDH, LS, RS, LD, RD]

       if  L1n & L2n = { }, Ln = Lp

       else Ln = propagation_rule([Lp, L1n, L2n])

else

     if  L1n & L2n = { }, Ln = default(n)

     else  Ln = propagation_rule([L1n, L2n])

Procedure prune(T, n)

   /* Tree representing the document, Determines if n
has to be removed from T */

For each c ∈ children(n) do prune(T, c)

if children(n) = { } and Ln ≠ '+'  then

   remove the current node from T
```

Existing XML access control techniques determine whether to allow a query to access or not after labeling the DOM tree. Thus the system has to keep all information necessary for right tests to the end unnecessary right tests were repeated [5]. Such extra tasks slow down the speed of access control.

4 Evaluation

```
<!DOCTYPE authorizations[
<!ELEMENT set of authorizations (authorization)+>
<!ELEMENT authorization (subject, object, ALTG, action, sign, type)>
<!ELEMENT subject (#PCDATA)>
<!ELEMENT object (#PCDATA)>
<!ELEMENT ALTG empty>
<!ELEMENT action empty>
<!ELEMENT sign empty>
<!ELEMENT type empty>
<!ATTLIST set of authorizations about CDATA #REQUIRED>
<!ATTLIST ALTG value(R, DSLG) #REQUIRED>
<!ATTLIST action value (read, write, create, delete) #REQUIRED>
<!ATTLIST sign value (+ | – ) #REQUIRED>
<!ATTLIST type value (L|R|LDH|RDH|LS|RS|LD|RD) #REQUIRED> ]>
```

Fig. 3. XML Access Sheet base DTD

Our processor takes as input a valid XML document requested by the user, together with its XML Access Sheet (XAS) listing the associated access authorizations at the instance level. The processor operation also involves the documents DTD. The processor output is a valid XML document including only the information the user is allowed to access. To provide a uniform representation of XASs and other XML-based information, the syntax of XASs is given by the XML DTD depicted in Figure 3.

The existing access control is the repetitive tree labeling process and DTD verification process consume a lot of memory for XML parsing and DOM tree search, which may degrade system performance.

Table 1. XML Documents Transformations by the security processes

Execution Processes	
Request	1) XML documents request
Security Processor	2) Parsing(DOM tree) 3) Tree Labeling 3) DTD Validation 5) Check for right to change DTD structure 6) Change DTD 7) Change documents structure 8) Unparsing
View	9) XML documents result

In the Table 1 above, if the existing access control technique is used as in the case that a user's authorization changes XML documents and structures, the following procedure is necessary [3], [5].

Step 1: User sends an access request.
Step 2: Parsing of XML documents to examine an operator's authorization.
Step 3: Labeling of authorization to DOM trees using information on access authorization.
Step 4: Determining if structure is changed in the stage of testing DTD.
Step 5: Conducting exchange operation if the DTD test identifies that operation leads to no structure change.
Step 6: Parsing to obtain new DOM trees as XML contents were changed after the operation.
Step 7: Testing authorization of insertion operation.
Step 8: Labeling authorization to DOM trees and testing DTD.
Step 9: An insertion operator is denied because it was shown to change DTD.
As seen above, the existing access control can make the system inefficient with the labeling process to assess authorization after each demand by a user and repetitive visits to DOM trees.

To the contrary, the suggested access control policy model can separate operators' collection into ALTG and thus prevent delay in complicated authorization assessment and responding.

5 Conclusion

In this paper, we suggested an access control policy model and tree labeling algorithm for XML documents. Action label group was defined to solve problems in efficiency while adding operators to the model during access. The existing access control generated XML documents into DOM trees according to a user's demand, identified the XML access control list, and removed nodes with access denied and provided only those with access permitted to a user.

However, the definition of the ALTG made it easy to manage information on access authorization and users and remove unnecessary parsing and DOM tree searching, consequently providing rapid access control. It has a disadvantage of making the system inefficient through the labeling process to assess authorization after each demand by a user and repetitive visits to DOM trees.

Further studies are necessary on access control that reflects each property in other applications using XML type.

References

1. A. Gabillon and E. Bruno, "Regulating Access to XML Documents", In Proc. IFIP WG11.3 Working Conference on Database Security, 2001
2. Document Object Model(DOM), Avaiable at http://www.w3.org/DOM/
3. E. Bertino, S. Castano. E. Ferrari, M. Mesiti, "Specifying and Enforcing Access Control Policies for XML Document Sources", WWW Journal, Baltzer Science Publishers, Vol.3, N.3, 2000.

4. E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, P. Samarati, "Securing XML documents," in Proc. Of the 2000 International Conference on Extending Database Technology(EDBT2000), Konstanz, Germany, March 27-31, 2000
5. E. Damiani, S. Vimercati, S. Paraboschi, and P. Samarati, "Design and implementation of an access control processor for xml documents". In proceedings of the 9th International WWW Conference, Amsterdam, May 2000.
6. IBM Tokyo Lab, "XML Access Control Language", 2000, Http://www.tr.ibm.com/projects/xml/xacl/xaclpec.html
7. Michiharu Kudo. Satoshi Hada "XML Document Security based on Provisional Authorization" CSS 2000, Athens, Greece
8. OASIS-XACMLTC, "OASIS eXtensible Access Control Markup Language", Working Draft 15, 12 July 2002, http://www.oasisopen.org/ommittess/xacml/repository/draft-xacml-schema-policy-15.doc
9. Sun's XACML Implementation. http://sunxacml. soureefore.net/.
10. T. Bray et al. "Extensible Markup Language(XML) 1.0". World Wide Web Consortium(W3C). http://www.w3c.org/TR/REC-xml(October 2000).
11. World Wide Web Consortium(W3C), "XML Path Language(XPath) Version 1.0", http://www.w3.org/TR/PR-XPath 19991008, (October 1999).

APPROXIMATING SAML USING SIMILARITY BASED IMPRECISION

Guillermo Navarro[1], and Simon N. Foley[2]

[1] *Dept. of Information and Communications Engineering,*
Universitat Autgnoma de Barcelona, 08193 Bellaterra, Spain
gnavarro@ccd.uab.es

[2] *Dept. of Computer Science,*
University College, Cork, Ireland
s.foley@cs.ucc.ie

Abstract With the increasing complexity of networked systems has come the trade-off of security versus functionality; a strictly secured system is often an unusable system. As a consequence, users often entirely bypass security in order to get their job done. We consider how similarity techniques that are used by case-based reasoning systems can be used to provide a degree of control over how strictly/precisely security is enforced. The flexibility to be able to meaningfully control how strictly security is enforced is especially relevant in the emerging Web Services architectures, where a wide variety of different users and heterogeneous systems use a common framework to interoperate with a wide variety of different resources and services. The paper proposes *similarity-based imprecision security* (SBIS) for the *Security Assertion Markup Language* (SAML) as an approach to managing security in a web-services environment.

Keywords: Imprecise security, SAML, Case-Based Reasoning, access control.

1. Introduction

Traditional research on protection systems has focused on finding a system that can provide *absolute security.* That is, systems where only properly authorised actions can take place. By properly authorised we mean that the actions need to be absolutely classified, identified, authenticated, an so forth. Modern computer systems have become very complex. Many users may wish to interact and/or use many resources, which may be distributed across a network. Enforcing security across these systems becomes correspondingly complex and, if strictly enforced, can lead to an unusable system.

End users regularly fail to appreciate the security decisions that they must make because they barely understand the security policy in force, let alone how security mechanisms may interoperate. Users tend to deliberately ignore or bypass security to get their work done. For example, it is a common practice for users to deliberately share or disclose passwords to facilitate system access (Adams and Sasse, 1999). Large enterprises such as governments, academic centres or big corporations, have many formal rules and regulations. In practice, enterprises work by relying upon social networks and unwritten rules, which are often contrary to the written rules. After all, a strategy used by employees to pressure management in labour disputes is to *work to rule* (Odlyzko, 2003).

An example of the potential for complex security rules is the *Secure Assertion Markup Language* (SAML) that is used to express security information in Web Services and Grid. The use of overly strict security policies by Web Services will more than likely lead to security being bypassed by administrators in their effort to provide continuing service. We use approximation techniques from the area of Case Based Reasoning to provide a degree of control over how strictly a security policy is enforced. Rather than the conventional all-or-nothing security, our approach can be regarded as providing a security 'dial' that controls the degree of strictness of security enforcement that the system is willing to tolerate.

In this paper we describe how these approximation techniques can be used in a practical way in SAML based applications, such as web services. Section 2 motivates and describes related work. In Section 3 we describe similarity-based imprecise security. Section 4 describes how it is integrated with existing SAML-based frameworks and Sections 5 and 6 describe the extension of SAML to support imprecision information in practice. Section 7 concludes the paper.

2. Motivation and Related Work

Empirical studies reveal that security systems are failing to provide usable applications, from simple password-based systems (Adams and Sasse, 1999; Yan et al., 2004), to complex access control systems (Zurko and Simon, 1996). Existing research on the usability of security systems has mainly focused on user interfaces. While providing a better user interface may be an excellent solution for some systems, it is not necessarily the solution to more usable and secure systems (Whitten and Tygar, 1999; Smetters and Grinter, 2002).

In this paper we propose the use of a different approach to achieve more usability in security systems, by reconsidering how security decisions are taken. We consider an *imprecise* security system, where the decision engine can take into account the similarity between security related information. *Imprecise se-*

curity introduces more flexibility in protection systems. Imprecision is defined in terms of similarity of authorisation: for example, how similar is 'root' access in Unix to 'administrator' access in Windows? Rather than all-or-nothing security, imprecision provides degrees of security, which can be viewed as a a *security dial.* Turning the dial up, results in the system becoming closer to a very strict security system (in some sense more secure); turning the dial down relaxes security enforcement, permitting more imprecision (in some sense less secure). Some applications of *imprecise security* are:

- *Overcome complexity in large organisations.* Large organisations impose many administrative rules which can be counterproductive; employees deliberately bypass the rules to do their job. An absolute security system does not allow its employees to bypass the rules. To avoid bureaucracy, users stop using the system whenever possible, or else use it incorrectly (for example, sharing passwords or private keys).

- *Emergency situations.* Some emergency situations may require relaxing the security measures of a system. This security downgrading should be achieved in a fast and controlled way. For example, a doctor isolated in an hospital during a tropical storm, needs to access the records of a patient from another department, doctor or hospital.

- *Heterogeneous systems.* Interoperability of heterogeneous systems require the use of security information from one system to another, or reuse security policies between different environments. Due to the different nature and technologies of the different systems it may be impossible to provide an isomorphic mapping between them. Similarity measures provide degrees of imprecision that can allow a practical mapping to be defined.

Providing degrees of flexibility in security enforcement has been studied to a limited extent in the literature. In (Rissanen et al., 2004), the authors introduce the notion of *override* in access control policies. An access control request can be denied with the possibility of override. If the user agrees, the system allows the access under the audit of some authority. Our approach differs from this by providing, in effect, greater flexibility in defining how authorisations may be overridden. (Povey, 2000) proposes an *optimistic* security model that assumes that every access is legitimate and should be granted under the basis that the system can rollback illegitimate actions. We believe that in practice it would not be feasible to undo all illegitimate actions, and some minimum security should be provided. Nevertheless, while not following the dictum "Make the user ask for forgiveness not permission" (Blakley, 1996), our proposal does adopt some optimistic security model principles (see Section 3).

The research described in this paper builds on the suggestion in (Foley, 2002) that similarity measures could be used to provide imprecision in delegation for trust management systems. Supporting imprecision for information retrieval systems has been extensively considered in the literature on similarity-based retrieval for Case-Based Reasoning (CBR) systems (Aamodt and Plaza, 1994). Imprecision permits answers that may not formally meet the query condition, but can be considered 'close enough'. The contribution of this paper is a consideration of how imprecision techniques can be used in a practical setting, and in particular, how similarity can be usefully introduced to SAML and supported within Web Services and Grid architectures.

3. Similarity-based Imprecise Security Systems

We define a *similarity-based imprecision security* system or *SBIS* system, as a security system, where the security decisions take into account the similarity between permissions, attributes, authorisations, or other security-related information.

SBIS Characteristics

In general, a system that supports SBIS has the following characteristics.

- *Accountability*: given the imprecision supported by the system, there needs to be some guaranties of accountability. In SBIS, accountability may be achieved by strong authentication of the principals. Whether this authentication is provided by means of a centralised authority such as a PKI or not, will depend on the specific scenario, environment, and configuration of the system.

- *Auditability*: in order to provide accurate postmortem analysis of the system's operations. All operations permitted under similarity constraints should be logged in detail, so they can be analysed to study possible irregularities.

- *Constrained entry points*: as stated in (Povey, 2000), exceeding privilege should be a rarity, rather than a norm. If most of the actions need to be checked through the SBIS security check because they do not pass the absolute security check (see Figure 1), it is a symptom that the system is not properly set up.

- *Deterrents*: as in many access control systems, it is interesting to have mechanisms to punish principals who misbehave. This punishment can be economic or simply, restricting access permissions during a period of time (recall, "Make the user ask for forgiveness not permission").

- *Least intrusive*: one of the main ideas behind SBIS, is for it to be easy to integrate within existing security systems. SBIS systems can use any type of security related information, permissions, authorisations, security policies, rules, and so forth, as long as a similarity function is provided between them.

Similarity

Similarity is equivalent to the dual distance concept from a mathematical point of view, and has been successfully applied in CBR systems. There are several similarity function types and families, for instance, there are boolean, numeric or partial order (including *lattices*) functions. Boolean functions may be easily emulated with numeric functions if required, and a variety of numeric similarity values can be expressed or normalised to the domain $[0, 1]$, including $\mathbb{R} \cup \{\ \infty, +\infty\}$. For the sake of simplicity, we consider only numeric similarity values. For a good review of lattice based metrics see (Osborne and Bridge, 1997).

Broadly speaking, a similarity function (denoted as '$\sim$') takes two arguments from a set of features P (permissions, attributes, etc.) and returns a similarity value:

$$\_ \sim \_ : (P \times P) \rightarrow [0, 1]$$

Similarity functions are reflexive ($x \sim x = 1\ \forall x \in P$), and symmetric ($x \sim y = y \sim x\ \forall x, y \in P$).

The similarity function is applied to a concrete feature or to a set of features. In the CBR literature, there are some generic similarity functions such as the weighted nearest neighbour, induction, lattice based metric, or the similarity matrix. In some cases one can compose functions or create specific ones.

Table 1. Sample similarity matrix.

$\_ \sim \_$	*und*	*phd*	*prof*
und	1.0	0.7	0.2
phd	0.7	1.0	0.5
prof	0.2	0.5	1.0

For example, Table 1 defines a simple similarity matrix between three roles: undergraduate student (*und*), PhD student (*phd*), and professor (*prof*).

Similarity threshold

The *similarity threshold* is the threshold for which a given similarity value is accepted. We denote the similarity threshold as 'δ'. For instance, suppose an SBIS system where some action requires a permission p, but the user does not hold such permission. The SBIS engine should look for a permission p' held by the user such that it is similar to p with a similarity value greater than the similarity threshold: $\exists p' | p \sim p' \geq \delta$.

The similarity threshold can be either *static* or *dynamic*. An *static similarity threshold* cannot be changed during execution time, while a *dynamic similarity threshold* can change its value during execution-time giving cause for a *security dial*.

4. Integration of SBIS in current SAML frameworks

An interesting issue is how an SBIS system could be integrated into an existing access control system, without having to introduce major modifications in the system. Our approach is to reuse the main components of an existing system. For instance, permissions, authorisations, attributes, and so on as provided in SAML assertions. We consider the use of SAML in a generic access control scenario as described in Figure 1. It involves a *Policy Decision Point* (PDP), which can take SAML assertions as the input to the access control decision, and a *Policy Enforcement Point*, which controls the access to a given service or resource.

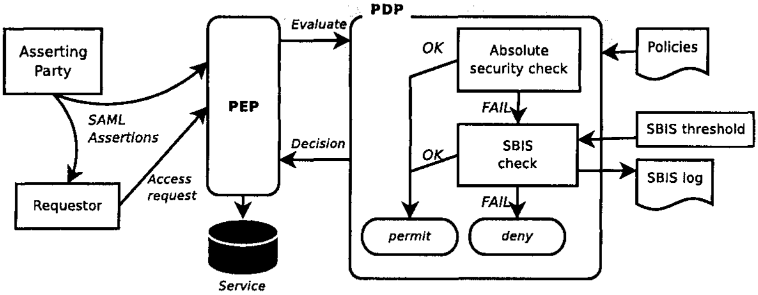


Figure 1. SBIS

Consider a set of SAML assertion statements P, with a partial order $\preceq$. Suppose that a given access request requires permission p and that the user making the request holds a set of statements Q. Then, the decision process could be summarised as:

Absolute security check: if $\exists q \in Q \mid q \preceq p$ then *permit*, otherwise SBIS check.
SBIS check: if $\exists q \in Q \mid q \sim p \geq \delta$ then *permit*, otherwise *deny*.

5. Expressing SBIS in SAML assertions

SAML provides a standard XML framework for exchanging security information between online business partners (OASIS, 2005). It is a well known standard applied in numerous industry products.

Security information is exchanged in form of *assertions*. SAML provides three types of *assertion statements*: Authentication, Attribute, and Authorisation Decision. Broadly speaking an assertion has an *issuer*, a *subject* or *subjects*, some *conditions* that express the validity specification of the assertion, and the *statement* (one or more). The assertion may be signed by the issuer.

Similarity-based assertion

We apply similarity functions to the SAML *statements* and their elements. For example, an authorisation decision includes the *resource* and the *action* of the authorisation. An authority may provide an authorisation decision assertion in terms of similarity between resources or actions, under some similarity threshold.

We introduce a new optional element contained by the SAML *statement* element called `SbisInfo` to provide SBIS-related information. Note that this information cannot be provided in the *condition* element of the assertion because it makes reference to the whole assertion, while a single assertion can provide more than one statement for the same subject.

```
<element name="sbis:SbisInfo" type="sbis:SbisInfoType"/>
<complexType name="sbis:SbisInfoType">
  <sequence>
    <element name="sbis:threshold" type="double"/>
    <element name="sbis:source" type="ID" minOccurs="0"/>
    <element name="sbis:function" type="sbis:SbisFunctionType" minOccurs="0"/>
    <element name="sbis:history" type="sbis:SbisHistoryType" minOccurs="0"/>
  </sequence>
</complexType>

<complexType name="sbis:SbisFunctionType">
  <sbis:element ref="cbml:similarity"/>
</complexType>
<complexType name="sbis:SbisHistoryTipe">
  <element ref="saml:AssertionURIRef" maxOccurs="unbounded"/>
</complexType>
```

Figure 2. SbisInfo XML Schema definition.

This *SbisInfo* element contains a sequence the following elements:

- *threshold*: the value of the threshold when the assertion was declared. If it is numeric, it will have to be normalised to the interval $[0, 1]$.

- *source*: optional element indicating the source of the similarity check, which is used to decide whether to trust the assertion or not. It will normally be the issuer of the assertion but this may not always be the case.

- *function*: the function used to calculate the similarity. This function is described using CBML (*Case Based Markup Language*) (Hayes and Cunningham, 1999), as described in Section 3.

- *history*: a sequence of URI references to assertions used to evaluate and issue this assertion. They may be used to verify the similarity constrains by a third party.

A simplified W3C's XML Schema definition of the *SbisInfo* element can be seen in Figure 2, which provides an extension of the current SAML assertion Schema (version 2.0). There, we denote the SAML namespace as *saml*, the CBML namespace as *cbml*, and the SBIS one as *sbis*.

```
<Assertion
    xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    Version="2.0"
    ID="http://www.library.edu/AuthenticationService/SAMLAssertions/126"
    IssueInstant="2005-04-13T16:30:00.173Z">
  <Conditions
    NotBefore="2005-04-13T16:30:00.173Z"
    NotOnOrAfter="2005-04-14T16:30:00.173Z"/>
  <Issuer>
    http://www.library.edu
  </Issuer>
  <Subject>
    <NameID NameQualifier="http://www.library.edu">
      Alice
    </NameID>
  </Subject>
  <AuthzDecisionStatement
      Resource="http://www.library.edu/PhDCatalog"
      Decision="Permit">
    <Action>read</Action>
    <SbisInfo>
      <threshold>0.5</threshold>
      <source>http://www.library.edu</source>
      <function>...</function>
      <history>...</history>
    </SbisInfo>
  </AuthzDecisionStatement>
  <ds:Signature>...</ds:Signature>
</Assertion>
```

Figure 3. SAML Assertion example.

Figure 3 shows an example of a simplified SAML authorisation assertion with SBIS information.

Expressing the similarity function

In order to express the similarity function, we use CBML, which is a generic XML-based language for CBR. This language can describe generic similarity functions (Coyle et al., 2004).

It is important to note the relevance of being able to express similarity functions in a common and standard way. An authority which provides some kind of attribute, can also provide the similarity function for its attributes. The authority will be the principal with more knowledge about the attribute, thus the expert who can provide the best similarity function. The similarity function, also serves as a proof to third parties of how the similarity was calculated. It includes the features used to calculate the similarity and how were they calculated.

6. Practical considerations

When a SAML asserting party or authority generates an assertion under similarity constraints the assertion includes the relevant information regarding the similarity threshold, similarity function and references to assertions used during the decision process. This information, together with the digital signature of the assertion is crucial for third parties that have to evaluate the assertion.

The SBIS information encoded within the assertion is sufficient to avoid the cascading problem (Foley, 2002). For example, given the example of the similarity function in Table 1, for the roles: undergraduate student (*und*), PhD student (*phd*), and professor (*prof*). Consider that Alice has an attribute assertion issued by a recognised authority *Chancellor*. If we consider a decision point with a similarity threshold $\delta = 0.5$, Alice will be able hold the *phd* role ($und \sim phd = 0.7$), but not the *prof* role ($und \sim prof = 0.2$). In some situations Alice may ask the decision point to issue a new assertion stating that she can hold the role *phd* to use the assertion in another access request for example. But note that then, she could use such assertion to gain privileges for the role *prof*, since $phd \sim prof = 0.5$. While this is a simplistic example, the cascading problem must be dealt with carefully in more complex situations. To avoid cascades, when Alice asks the decision point to issue the second assertion, this assertion will have all the information previously commented. Thus, the receiver is able to evaluate and track the similarity constraints used to issue the assertion.

In (Foley, 2002) a specific solution for the cascading problem is provided. While providing an elegant solution, it requires an *a priori* knowledge of the similarity functions involved in the similarity-based decision (including future ones). In this paper we provide a more generic solution. One could use the assertion to compare it to a new attribute assertion (such as age) and provide a new different similarity function.

7. Conclusions

This paper discusses the relevance and motivation for allowing a controlled degree of imprecision in security decision engines. This degree is based on the similarity between security related information such as permissions, attributes, etc. resulting in *similarity based imprecision security* (SBIS) systems. We introduced SBIS capabilities in SAML, which is a widely adopted standard in Web Services and Grid.

SBIS is not intended for high security or critical systems but systems where usability is a key point. Empirical studies demonstrate that currently, absolute security systems are failing to do their job. SBIS can provide enough flexibility to the system and at the same time it can ensure some degree of security.

Acknowledgments

The work of G. Navarro has been partially funded by the Spanish Ministry of Science and Technology (MCYT) though the project TIC2003-02041.

References

Aamodt, A. and Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AICom- Artificial Intelligence Communications*, 7(1).

Adams, A. and Sasse, M. A. (1999). Users are not the enemy. *Commun. ACM*, 42(12).

Blakley, B. (1996). The emperor's old armor. In *Proceedings of the 1996 workshop on New security paradigms.*

Coyle, L., Doyle, D., and Cunningham, P. (2004). Representing similarity for CBR in XML. In *Advances in Case-Based Reasoning (Procs. of the Seventh European Conference).*

Foley, S. N. (2002). Supporting imprecise delegation in keynote using similarity measures. In *Proceedings of International Security Protocols Workshop.*

Hayes, C. and Cunningham, P. (1999). Shaping a CBR view with XML. In *Proceedings of the Third International Conference on Case-Based Reasoning and Development, ICCBR-99.*

OASIS (2005). Assertions and Protocols for the OASIS Secure Assertion Markup Language (SAML) v2.0. sstc-saml-core-2.0-cd-04, Committee Draft 04.

Odlyzko, A. (2003). Economics, psychology, and sociology of security. In *Financial Cryptography: 7th International Conference.*

Osborne, H. and Bridge, D. (1997). Models of similarity for case-based reasoning. In *Procs. of the Interdisciplinary Workshop on Similarity and Categorisation.*

Povey, D. (2000). Optimistic security: a new access control paradigm. In *Proceedings of the 1999 Workshop on New Security Paradigms.*

Rissanen, E., Firozabadi, B. Sadighi, and Sergot, M. (2004). Towards a mechanism for discretionary overriding of access control. In *12th International Workshop on Security Protocols.*

Smetters, D. K. and Grinter, R. E. (2002). Moving from the design of usable security technologies to the design of useful secure applications. In *Proceedings of the 2002 Workshop on New Security Paradigms.*

Whitten, A. and Tygar, J. D. (1999). Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *Proceedings of the 8th USENIX Security Symposium.*

Yan, J., Blackwell, A., Anderson, R., and Grant, A. (2004). Password memorability and security: Empirical results. *IEEE Security & privacy*, 2(5).

Zurko, M. E. and Simon, R. T. (1996). User-centered security. In *Proceedings of the 1996 Workshop on New Security Paradigms.*

A DISTRIBUTED PROXY ARCHITECTURE FOR SERVICE DISCOVERY IN PEER-TO-PEER NETWORKS

Marcos Madruga[1], Thais Batista[2] and Luiz Affonso Guedes[3]
[1]Federal University of Rio Grande do Norte, CT- DEE, Campus Universitario - Lagoa Nova,59072-970 - Natal - RN – Brazil, madruga@unp.br; [2]Federal University of Rio Grande do Norte, CCET -DIMAp, Campus Universitario - Lagoa Nova,59072-970 - Natal - RN – Brazil, thais@ufrnet.br; [3]Federal University of Rio Grande do Norte, CT-DCA, Campus Universitario - Lagoa Nova,59072-970 - Natal - RN – Brazil, affonso@dca.ufrn.br

Abstract: In this work we present a service discovery system that supports flexible queries using partial keywords and wildcards. It is built upon a Chord network and it guarantees that any existing data that match a query is found. The main feature of this service is to use a proxy server layer with a mechanism for data distribution that reduces the number of nodes involved in the searching process.

Key words: service discovery, peer-to-peer, proxy, Chord, distributed searches.

1. INTRODUCTION

With the phenomenal success of Internet and the availability of a great amount of information, one of the main challenges nowadays is to find specific information. This problem was first noticed when it started to be hard finding specific information on web sites. This made sites like Google and Yahoo, that search the web for documents and other information, to become very popular. However, with the power of distributed computing and its different paradigms, the problem of finding components that provide a given service is going to a new dimension.

Several protocols have been proposed and we can generally divide them in two groups: (1) protocols developed to be use in limited environments, like local area networks such as SLP (Service Location Protocol) [12], UPnP (Universal Plug and Play Protocol) [13], Jini [14], and others; (2) a new series of protocols [5, 6, 7, 8, 9] developed to work in large scale, for instance those built on Peer-to-Peer networks [1, 2, 3, 4].

In this work we present a system that fits into the second category and uses a Chord [2] network as its foundation. It allows users to search for information by specifying keywords and wildcards. It also guarantees that existing data are found. The main feature of our system, when compared to others, is that it uses a proxy structure to accelerate searches and define a scheme to decrease the number of nodes searched for potential matches. This schema seams efficient even when the user gives little information to be used in the searching process.

This paper is structured as follows. Section 2 presents the background of the work that consists of briefly presenting Chord network. Section 3 presents the proposal of this work including the idea of distributed proxies and the changes we suggest in Chord. Section 4 presents the searching protocol. Section 5 comments about related work. Finally, section 6 presents the final remarks.

2. BACKGROUND

Chord [2] is a distributed lookup service used in peer-to-peer systems. It is not a storage system. It is based on the notion of consistent hashing and of an identifier space that is mapped to a set of nodes. A node is a process or a host identified by an IP address and a port. A chord identifier is associated with each node. Thus, high level names are translated into chord identifiers.

This work is built on a Chord Network, which is a peer-to-peer network and uses DHT (Distributed Hash Table) and consistent hashing [10, 11] techniques, to associate an identifier to each search key and each IP address of the nodes of the network. This is done through a hash function, like SHA-1, for instance, to the key and the IP address. It organizes the network in a ring layout, that is, to each key it is possible to determine the IP address of the server that contains it. In short: suppose that <hash(key)> is equal to 45. This means that this key will be on the server that has a IP address which hash function(IP) >= 45. When there is no server with hash(IP) = hash(key), the server related to that key will be the next in the ring that has an hash(IP) greater than that key.

To identify the next servers in the ring, each server provides a type of *router table*. The server localization contains at most O(Log N) messages traded by servers. In addition, the protocol supports the insertion and removal of nodes in the network and allows warning this to any application responsible for implementing the key migration to the new server (just the ones it is responsible for).

3. DISTRIBUTED PROXIES

Although the DHT (Distributed Hash Tables) based data publishing and localization schemes, as in the Chord Net, are already capable of distributing data between several servers, we use a technique that reduces dramatically the load on each of them through the insertion of a *proxy layer* between the nodes that query for data and the nodes that store them. Each proxy caches the query and serves several clients. Thus, instead of the nodes receive queries of N clients, they receive just from M proxies, where, of course, M is far less than N.

The distinguishing features of the proposed architecture are: it is automatic (no configuration is necessary to point which proxy to use), highly distributed (each node acts as a proxy) and each proxy takes charge of just a subset of keys, providing a greater rate of cache hit.

3.1 Chord Subnets

The model we propose is based on the creation of several Chord subnets inside the Chord global net. These subnets, however, are just logical subnets, as they are composed of the same machines of the global network. This idea, of course, requires changes in Chord in order to support that the same node be part of two networks (one of subnets and also the global network). These modifications are basically to duplicate the structure of the data that contain information about the nodes (and the keys) in the Chord ring, and to create an identifier for each network, that will be provided in the operation made through it (the network). The identifier will indicate which set of data structure should be used to determine the nodes that will be used.

Figure 1 shows a situation where there are three Chord subnets inside the Chord global network (containing nodes 1, 3, 6, 7, 10, 11, 15,19, 20 and 24) that allows 24 different keys. The first subnet has s1 as identifier and has nodes 1, 7 and 10. The second subnet has s2 as identifier and contains nodes 3, 11 and 19. Finally, the third subnet has s3 as identifier and contains nodes 6, 15, 20 and 24. Figure 2 shows the nodes and their related keys in the third subnet and in the global network.

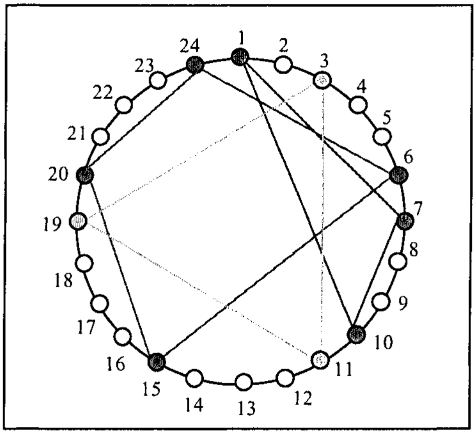


Figure 1. Chord Subnets.

Node	Keys	Node	Keys
1	1	11	11
3	2-3	15	12-15
6	4-6	19	16-19
7	7	20	20
10	8-10	24	21-24

Node	Keys
6	1-6
15	7-15
20	16-20
24	21-24

a)Global Network b) Third subnet: s3

Figure 2. Key distribution in the Chord Networks: Global Network and Third subnet

Each Chord subnet (sub ring) works as a proxy network (each node is a proxy). When a node is looking for data related to a specific key, it sends the query initially to the nodes of the Chord subnet to which it is part of, and these nodes (the proxies) are the ones that search for the data in the nodes of the global network. It is important to note that the nodes of the subnets store just the data associated to the keys in the subnet, as these indicate the set of data to each the proxy should be used.

3.2 Proxy and Operation Mode

The proxy identification (subnet node) to be used in the searching process for a specific key is done in the same way as in any Chord network. The only different aspect is the fact that the subnet identifier must be informed, so that the nodes are able to know whether they should use the data structure (that determine the next node) related to the global network or to the subnet. Also, we should notice that length of the key and hash function used are the same, for both subnet and global network. To use the same key length is possible due to the Chord architecture, which allows the responsibility of a set of N keys to be distributed in a network with any number of physical nodes. Also, the use of the same hash function simplifies and optimizes the system performance, as it allows calculating this function only once, with the proxy just forwarding the key search in the global network.

When a proxy server receives a query for a specific key with the network identifier being the local subnet, it is changed to the one of the global network and it forwards the query to the next node of the global network. This is done after searching its own cache to check if it already has that value stored locally.

Suppose that node 24 in Figure 1 is trying to find key 9. Initially it searches, in the local network (s3): it checks its subnet table (see Figure 2b) and determines that the node (proxy) responsible for that key is node 15. Thus, it forwards the query to that node and informs that subnet s3 is being referred to, that it is an operation inside the subnet. When node 15 receives the query, it changes the Chord net identifier from subnet to global network. It verifies in the global network table (see Figure 2a) that node 10 is responsible for key 9, and forwards the query to that node (in case it does not have the data related to this key in its cache). When the query is received, node 10 is informed by the network identifier that it is a global search. Thus, it will not act as a proxy, but as a normal node just recalling the data associated to the informed key of its local database.

3.3 Performance

It is well known that the performance of a Chord network is influenced by its capacity to determine the IP address of the node responsible for a specific key. This is done by contacting at most O (Log N) nodes, whereas N is the number of nodes in the network. However, as each proxy searches for just a one subset of the total keys, we can use an efficient cache system for the IP addresses of the nodes responsible for each key. In other words, each proxy besides caching the searched keys and the data associated to each of them, it also caches IP addresses of the nodes associated to each of them. This way we reduce the complexity of the searches from O(Log N) to O(Log S), where S is the number of nodes in the subnet. Thus, it is far less than N.

4. SEARCH PROTOCOL

A Chord network is able to process simple key searches. It is possible to determine in which node a given identifier is stored. This way it cannot be used directly as a protocol for complex searches with detailed descriptions of services. Therefore, usually a Chord network (or other peer-to-peer networks, like CAN [3]) is used as basis for building refined search protocols. Some desirable features in such protocols are: (1) Ability to distribute the registry of services, even if they are of the same kind, to different servers. This avoids demanding services to end up overflowing specific nodes; (2) Load balance, so that the loading of performing searches is distributed between each server; (3) Exact matches of complete sentences must be found in a very precise way; (4) Guaranteed success in the search if the data is available in the network.

Services can be described by a pair (attribute, value) or by a XML document. In this work we will use the former, and not the XML approach, even though they are very similar. Also, we assume that a service description has an attribute identifying each service type, for example: printer.

4.1 Service Registry

The service registry process consists of the storage of each attribute of the service in a different node, together with the service type identifier and a link (an URL, for instance) to the complete description of the service. Each link should identify the service in a unique way, as it will associate the various attributes of the service.

Although the attributes are registered separately, complex searches dealing with various attributes are also possible by splitting the search into several simple searches. The simple search deals with just one of the attributes. After, the results are grouped according to the operators used (OR, AND and so on). Another way of searching, that is more reliable for searches dealing with the operator AND, is to send the search to the node responsible for one of the specified attributes. From this node each one selects the services that match the specified criteria and forwards the search to the node responsible for the next attribute. The services found would be forwarded from node to node are refined at every new step.

The identification of a node where an attribute should be stored is calculated in the following way:

1. It is calculated the hash function of the text resulted from the concatenation of the name of the type of service and the name of the attribute, that is: hash(kind_service+name_attribute).

2. It is calculated the numeric value for the data in the attribute, using table 1, where each character is associated to a value. The calculation consists of adding the numeric value of each character of the text.
3. For each word in the attribute, we search for the character with the least numeric code associated and subtract this value from the numeric code associated to the character with greatest value. The biggest value is used.
4. It is divided the keys from the one obtained in step 1 in intervals of *n* keys, where *n* is the code of the greatest value in table 1.
5. The value obtained in step 2 is added to the key calculated in step 1 creating, then, a new key.
6. It is verified to which of the intervals calculated in step 4 the obtained key is part of. The data will be stored in the node *m* of this interval, where *m* is the value calculated in step 3.

Table 1. Table of characters codes

Code	***Character***	***Code***	***Character***	***Code***	***Character***
6	A	8	C	10	E
7	B	9	D	...	...

Figure 3 shows the process of identifying the node responsible for the *color* attribute of a service called "*car*". Note that the hash function result is *node 3* and that adding to this number the value obtained by the sum of the numeric code of all characters of the attribute value, that is, black, we obtain the value 8 is result. Assuming that the greatest code in the characters table is 4, the nodes from node 3 obtained through the hash function, are divided in intervals of 4 nodes. Whereas, interval 1 has nodes 3, 4 ,5 and 6. Interval 2 has nodes 7, 8, 9 and 10.. As node 8 is in the second interval the data will be stored in any of its nodes. The exact node of the interval is determined by the difference between the codes of the characters 'k' and 'a' obtained from the value "black". As for our example we assumed that this difference is 4, the resulting node is node 10.

4.2 Queries

When a search for an attribute is composed of the complete value of the data in the attribute, we just need to make calculations identical to those done in the registry process, already explained, in order to obtain the specific node where the attribute is located. Note that, this approach can guarantee the requirement that exact searches must be as efficient as possible.

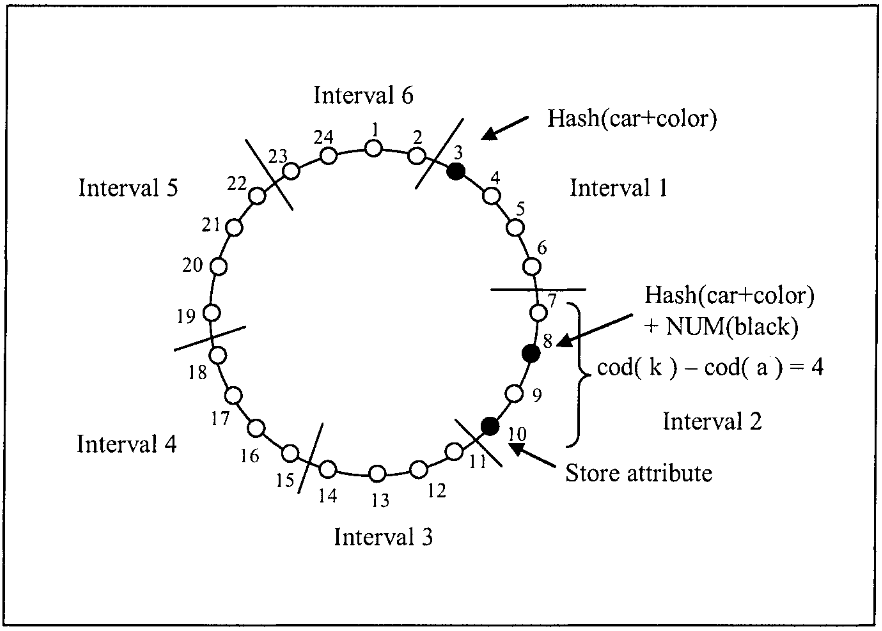


Figure 3. A identification of the node responsible for storing the value of the attribute "color=black" for a "car" service. We assume that the sum of the codes of the characters of "black", NUM(black), is equal to 5 and that the difference between letters 'k' and 'a' (greatest and least characters of "black") is equal to 4.

However, in a partial search, searching for keywords or expressions like comput*, for instance, does not return the specific nodes where the data are. The motivation of using a method that ascribes a numeric value to the data in the attribute through the sum of the codes in its characters is that this places a substring in every node right before the one responsible for the complete string. Also, as words characters are given by the user, it will be closer to the node containing the data. Therefore, in a partial search the results of the calculations inform the node where the search must start from, but the subsequent nodes in the Chord ring must also be queried.

This composition of nodes can be very big. In order to address this problem, a technique was developed to divide the nodes after the base node (pointed by the application of the hash function) in intervals and use the difference between the codes with greatest and least character of the data to help determine the exact node where the data is. Our goal is that if just part of the complete text contained in the attribute is informed, the difference between the greatest and the least character in the informed text will always be less or equal to this difference calculated in the complete text. Thus, for each range from the one containing the node calculated as initially, just the node above the value of the difference between the greatest and the least characters informed in the search must be searched.

4.3 Optimizations

The number of the intervals to be searched can also be reduced if it is identified the data that creates the greatest numeric value possible (sum of the characters codes) for the searched attribute. To get this information it is necessary to use the length of the attribute field. To handle this issue, we concatenate to the searched data as many characters as necessary to hit the maximum length of the field. The character used should be the character with the greatest numeric value associated (see table 1). To determine the field length, the proxies should analyze the information obtained for the attribute and use the greatest data size already recalled, to calculate the maximum size.

Although this optimization provides the results expected from most of the searches, some data may not be found. Therefore, a mechanism should be provided to allow the user activate or not this optimization.

5. RELATED WORKS

Squid [8] is a searching system that also supports searches by using keywords and wildcards. It uses a space filling curve based on an index scheme to map data elements to nodes using keywords. The main features of our system is that it requires some keywords to both describe the service and the search, while it keeps its efficiency even with a small amount of information provided by the user. In addition, even though systems provide a fairly similar mechanism (based on the association of the numeric codes) to determine the set of nodes to match the query, our approach is different.

The INS/Twine [5] system registers services by calculating a hash function that involves both the field name and the given data. This calculation is processed several times (for the different fields of the service). Then, it stores the complete description of the service in each node. This approach decentralizes the services register but allows just exact searches. Our approach also uses the idea of registering the service based on the value of its attributes, but it registers just the data related to the attribute in each server and modifies the way that the hash function is used. Thus, partially searches can be made.

6. CONCLUSIONS

In this paper we have presented a service localization mechanism built upon a Chord network. However, it proposes some changes to Chord in order to support a proxy server layer, containing the network's own nodes, that cache the queried data in order to accelerate the searching process. The proxy architecture allows the automatic identification of the proxy to be used and it also supports fault tolerance. Another important contribution is a service distribution method between the nodes in the network, that allows the dramatically reduction of the number of potential matches in a search.

References

1. P. Rowstron and P. Druschel.: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. Lecture Notes in Computer Science, 2218, 2001.
2. A.Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: Scalable Peer-To-Peer lookup service for internet applications.In Proc. 2001 ACM SIGCOMM Conference, pages 149–160, 2001.
3. S. Ratnasamy et al.,"A Scalable Content-Addressable Network," Proc. ACM SIGComm, ACM Press, 2001, pp. 161–172.
4. C. Plaxton, R. Rajaraman, and A.W. Richa: "Accessing Nearby Copies of Replicated Objects in a Distributed Environment," Proc. ACM SPAA, ACM Press, 1997, pp. 311–320.
5. M. Balazinska, H. Balakrishnan, and D. Karger. INS/Twine: A scalable peer-to-peer architecture for intentional resource discovery. In Proceedings of the First International Conference on Pervasive Computing, pages 195–210, Zurich, Switzerland, August 2002. Springer-Verlag. College, February 2002.
6. D. Spence and T. Harris. Xenosearch: Distributed resource discovery in the xenoserver open platform. In *Proc. of HPDC 2003*, Seattle, WA.
7. C.Tang, Z. Xu, and M. Mahalingam, PeerSearch: Efficient Information Retrieval in Peer-to-Peer Networks, tech. report HPL-2002-198, HP Labs, 2002.
8. C. Schmidt and M. Parashar, Enabling Flexible Queries with Guarantees in P2P Systems, - Internet Computing Journal, Vol. 8, No. 3, May/June 2004
9. R. Huebsch, J. M. Hellerstein, N. Lanham, B. T. Loo, S. Shenker, and I. Stoica. Querying the internet with pier. In Proc. of the 29th International Conference on Very Large Data Bases, September 2003.
10. D. Karger, E. Lehman, F. Leighton, M. Levine, D. Lewin, and R. Panigrahy. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing* (May 1997), pp. 654–663.
11. D. Lewin. Consistent hashing and random trees: Algorithms for caching in distributed networks. Master's thesis, MIT, 1998. Available at the MIT Library, http://thesis.mit.edu.
12. E. Guttman. Service location protocol: Automatic discovery of IP network services. *Internet Computing*, July/August 1999.
13. UPnP Forum: Understanding Universal Plug and Play: A white paper. http://upnp.org/download/UPNP_UnderstandingUPNP.doc (2000)
14. S. Microsystems. Jini architecture specification, December 2001.

DIRECTIONAL ANTENNA BASED PERFORMANCE EVALUATION OF 802.11 WIRELESS LOCAL AREA NETWORKS

Kartikeya Tripathi, Janise McNair and Haniph Latchman
Department of Electrical and Computer Engineering, University of Florida, Gainesville, Florida 32611

Abstract: In this paper, the use of directional antennae on access points in wireless local area networks to sectorize the coverage area is proposed. This is shown to be a simple way to significantly increase throughput while greatly reducing the occurrence of the hidden terminal problem. First, we discuss the current use of sectorization in various wireless networks, including cellular and ad hoc networks. Then, we present the details of the proposed architecture, describing the necessary enhancements to the current Access Point functionality and examining the tradeoffs of costs and infrastructure. Finally, we carry out a performance analysis to demonstrate the marked increase in throughput and efficiency achieved by the new technique.

Key words: *Wireless Local Area Network (WLAN); 802.11; sectorization; Access Point (AP); Performance Analysis*

1. INTRODUCTION

Over the last five years, wireless local area networks (WLANs) have become an increasingly popular choice for users seeking data services both at home and at the office. The reason for the success is that WLANs can support highly crowded, hot spot areas needing high data rates with a minimal investment in infrastructure, functioning as a wireless extension of the Ethernet. However, WLANs still suffer from reduced data rates due to unreliable wireless links, error control overhead, and overhead for medium access control. Though standards such as 802.11a/g are evolving to achieve higher data rates by using orthogonal frequency division multiplexing (OFDM) at the physical layer, the 802.11 MAC continues to rely on an omni-directional access point (AP) that runs one instance of the carrier sense multiple access with collision avoidance (CSMA/CA) protocol. This approach has become a bottleneck for high-speed data and multimedia applications because of the increased overhead required to coordinate user access. In fact, the overhead incurred by RTS and CTS decreases the actual throughput of 802.11 to almost half the throughput that can be achieved at the physical layer.

A common technique used to overcome this problem, i.e., to increase WLAN throughput in high traffic areas, is to use several co-located APs to serve one network in an extended service set (ESS). For example, since the 802.11b standard specifies three non-overlapping frequency channels, three APs can be co-located, each using a different channel, allowing three simultaneous instances of 802.11, and improving the overall throughput. Similar results can be seen with the 802.11a standard, which can provide up to 8 orthogonal channels. In fact, multiple co-located APs for 802.11a are necessitated due to the smaller coverage area of the same. Though this procedure can solve the problem to some extent, requiring multiple APs to serve one location is an inefficient use of hardware, and points to an inherent capacity ceiling. However, with small improvements in the software and the hardware, a single AP can use the idea of a simple router to significantly multiply the achievable bandwidth at the last hop wireless link.

In this paper, we propose a new sectorized architecture for infrastructure-based WLANs. In Section 2, we discuss the current use of sectorization in various wireless networks, including cellular and ad hoc networks. Then, in Section 3, we describe the proposed modifications, including necessary enhancements to current AP functionality, and an examination of the tradeoffs with respect to costs and infrastructure. Section 4 provides a description of the performance analysis used to determine the effectiveness of the new scheme, followed by the numerical results in Section 5. Finally, Section 6 provides some conclusions and a discussion of future work.

2. RELATED WORK

A detailed description of the 802.11 medium access control (MAC) protocol can be found in [1], [2]. Quality of Servive (QoS) guarantees for delay sensitive traffic and the enhancement of the throughput in the 802.11 protocol have been topics of intensive research. Most of the new ideas in this area, as in [3], try to suggest some improvement in the system performance by making variations in the traditional CSMA/CA scheme for medium access control. Similarly, many mobility models and measures have been proposed for simulating moving terminals in a generic wireless coverage area, as in [4]. The use of directional antennas for WLAN access points has been studied in [5] and [9] as a space division multiple accesses (SDMA) technique to increase the capacity of cellular and WLAN communication systems respectively. Cell sectorization using directional antennas has often been implemented in cellular voice networks to increase the frequency reuse factor at the cost of a decrease in trunking efficiency [6]. In [7], a new MAC protocol for ad hoc wireless LANs was proposed, which was based on the CSMA/CA scheme using directional antennas. This scheme uses the idea of directional RTS and CTS to allow simultaneous conversations, which do not interfere with each other, and hence improve the throughput. However, little work has been done on medium access control for sectorized access points in infrastructure-based wireless LANs.

3. SECTORIZING THE COVERAGE AREA

3.1 Enhanced AP Functionality

The AP in the current 802.11 topology is a layer-2 device, which has only two network ports: one to connect it to the wired Ethernet, and the other to connect it to the radio interface. Such a design is effective for a small office environment, or for home networking, both of which serve a small number of mobile units (MUs). However, as 802.11 becomes a data networking solution for larger organizations with a large number of MUs, and also the possible 4G wireless solution, the simple AP design loses its efficiency. We propose the design of 802.11 APs with one Ethernet port and multiple radio ports, where each radio interface has a semi-directional antenna, covering, say, 360 degrees in the vertical and 90 degrees in the horizontal. Each directional antenna can be configured on one of the orthogonal channels (3 in 802.11b) to allow the data in each sector to be transmitted and received separately on each antenna. The antennas can be placed in a spatially predetermined way to minimize co-channel interference. Figure 1 shows the sectorized AP configuration for four directional antennas.

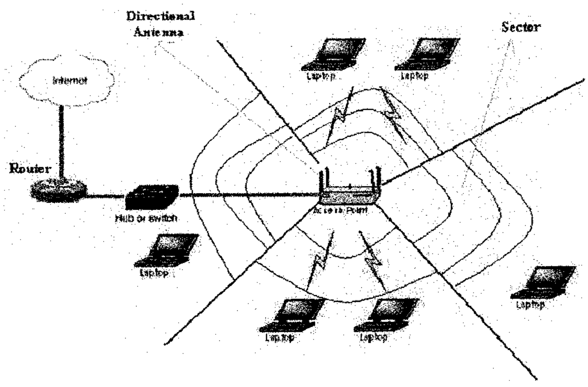


Figure 1. Sectorized AP Configuration for Infrastructured WLANs

3.2 Medium Access Control

We envision a separate MAC instance running in each of the sectors at the AP. The MUs in each sector contend amongst each other using CSMA/CA to gain access to their respective directional antennas. We do not expect buffering to be a major issue. For example, to run three instances of 802.11b at the AP, the Ethernet port will support 100 Mbps links, while each directional antenna can have a maximum rate of 11Mbps. We propose that the sectorized AP use a sector table, similar to a routing table, to route the downlink traffic efficiently. This table will comprise the current sector and the MAC address for each MU. As MUs move around the coverage area, this table will have to be updated every certain interval of time.

3.3 Trade-offs

In the implementation of the sectorized scheme, we expect to observe the following advantages:

- Since each single sectorized AP has multiple conversations taking place at the same time, the throughput can be increased.
- The hidden node problem will be reduced by sectorization, since there is a much smaller probability that MUs in the same sector cannot hear each other.
- An added benefit to the reduction of the hidden terminal problem is that the need for RTS/CTS overhead is eliminated, again increasing the throughput performance.

On the other hand, there are several issues to be considered for implementing sectorized APs, including:

- The new approach would significantly increase the computational complexity of the AP, which was originally built to be a simple device. However, considering the increasing demand for high bandwidth wireless services, the application of WLANs to high traffic areas, and the objective to provide QoS for multimedia traffic, the customer satisfaction resulting from increased throughput should offset the investment in added complexity.
- Similar to the above issue, the design of a new sectorized AP would be more costly for manufacturers and customers alike. However, considering the new technique as a replacement for the practice of using several APs in a single location, the reduced number of APs should offset the additional hardware cost. In addition, mass manufacturing of electronic components tends to reduce the cost in the long run.
- Having sectors would bring forth the issues of user mobility and its influences on the effectiveness on the system. In particular, as MUs move among various sectors, each cross over results in control signaling and updating of the AP's sector table, which takes away time from useful data processing. There are also fluctuations in the throughput of the system as the load in each sector varies, and this depends on the patterns of mobility of the users.
- Finally, if many MUs gather in a concentrated area, such as an audience in an auditorium or conference room, then there would be no activity in several sectors, while one or two sectors may be overloaded. Some cases are shown in figure 2. In this case, since the antennas are configured on different frequencies, each can be temporarily turned toward the problem sector to provide relief. Furthermore, a controlled feedback system can be designed to automatically change the orientation of the antennas, based on sector load and user density.

In light of the tradeoffs, and to quantify both the advantages and some of the pressing issues, we now evaluate the performance of the new technique.

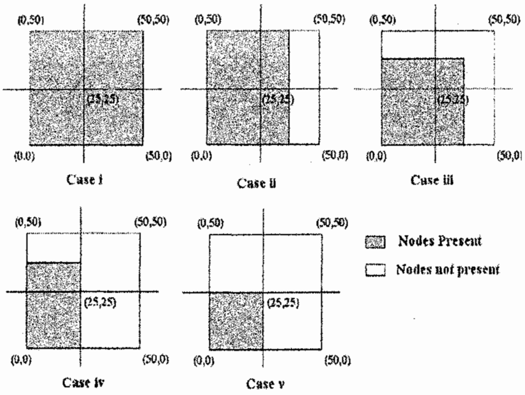


Figure 2. Variation in Node Density in a coverage area of 50x50 units

4. PERFORMANCE ANALYSIS

A discrete event analysis is used to analyze the performance of the scheme. We simulate the access behavior of N stations located in an LxL square unit area, with an S-sector AP located in the center.

4.1 Fading Model

The channel fading model used in test simulations is based on lognormal fading [8]. Log-normal model is a generic slow fading model. For this model, the path loss, L_P, in the signal propagation from node i to node j (and from AP to node) is given by:

$$L_{P,i,j} = L_O + 10\alpha \log d_{i,j} + X \qquad (1)$$

where L_0 is the path loss at the distance of 1 meter, α is the path loss exponent, $d_{i,j}$ is the distance between nodes i and j, and X is a random variable based on a log-normal fading distribution.

4.2 Hidden Node Calculation

A signal transmitted from one MU can be missed by another MU if it is so far away from the transmitter that the signal fades till it reaches the other MU. This can lead to collisions when the transmissions from 2 MUs hidden from each overlap. The signal fading is on the basis of the model described above. If the received power from node *i* to node *j* is less than a pre-set threshold, then nodes *i* and *j* are considered to be hidden from each other.

4.3 Collision Avoidance Calculations

The CSMA/CA protocol is used independently in each sector of the AP, along with the corresponding 802.11 back-off procedure [2]. Every node is assigned a contention window, based on a uniform distribution from a lower value of 0 to an upper value of CW_{min}-1 time slots. Each time a collision occurs, the contention window is doubled, up to a maximum of CW_{max}-1 time slots. A record is kept of the number of successful transmissions and number of collisions.

4.4 Throughput

The net throughput of the system is calculated in terms of the percentage of total simulation time used for successful frame transmissions:

$$Throughput = \frac{s \times t_{FRAME}}{T_{SIM}} \tag{2}$$

where s is the number of successful transmissions, t_{FRAME} is the transmission time of each frame, and T_{SIM} is the total simulation time.

5. RESULTS

5.1 Throughput and Hidden Nodes

Numerical results for the throughput and for the number of hidden nodes are obtained for the sectored coverage area, and are compared with those for the traditional 802.11 omni-directional AP. This set of results is obtained for fixed MUs, all of which are distributed uniformly in the composite coverage area (except in cases with different distribution densities). Table 1 shows the chosen parameters used in the simulation. The significance of the node density is to capture the impact of user positioning on the performance of the system.

Figure 3 shows the throughput results comparing the use of a three-sector AP with the omni-directional AP, for a varying number of nodes. The three-sector case has a rapid increase initially, due to the smaller number of nodes being distributed among different sectors, thereby reducing the number of collisions. As the number of nodes increases, the number of collisions also increases, resulting in a gradual throughput reduction for the three-sector case. Yet, even with the decrease, the three-sector throughput has a significant improvement upon the performance of the omni-directional case.

Table 1. Simulation Parameters

Parameter Description	**Value**
Path loss factor (α)	3
Square area (LxL)	L = 50 meters
Node Densities	Cases 1 – 5 (see Figure 2)
Slot time	50 micro seconds
DIFS	128 micro seconds
Packet Size	1 KB (constant)

Figure 4 shows the number of hidden terminal incidents for the sectored APs compared to the omni-directional case for the log normal fading model. It demonstrates that the number of hidden nodes is greatly reduced with an increase in the number of sectors used. As the coverage area per sector decreases, the MUs in a sector are closer and are less prone to being hidden from each other.

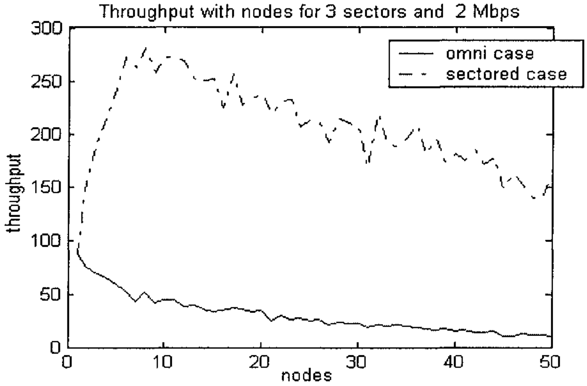


Figure 3. Throughput versus Total Number of Nodes (Density Case i)

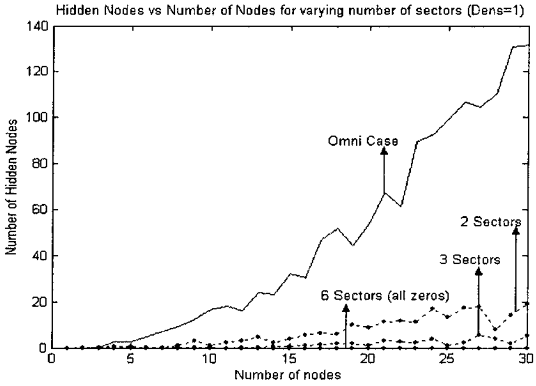


Figure 4. Number of hidden nodes vs. total number of nodes for Log-Normal Fading

The results in figures 3 and 4 were calculated for node density case i. As mentioned previously, one of the tradeoffs with the architecture is the effect of the node density on the performance of each sector. Figure 5 compares the throughput versus the number of sectors for various node densities, according to the cases illustrated in figure 2. It can be seen that the node density has minimal overall effect on the omni-directional throughput, while it has a drastic effect on the sectored AP throughput. The more concentrated the users are geographically, the more drastic the reduction.

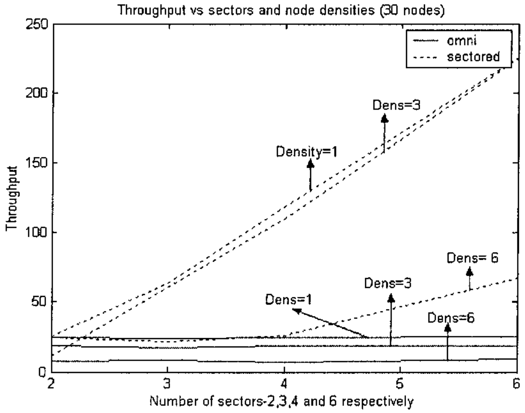


Figure 5. Comparison of Throughput vs. number of sectors for Log-Normal Fading

6. CONCLUSIONS

As the demand for wireless bandwidth increases with each new hot application or network, the efficient use of network resources, including hardware, becomes more and more important. In this paper, it was demonstrated that the use of sectorized APs in WLANs is an effective way to increase throughput as well as to prevent incidences of hidden terminals. The scheme coordinates multiple instances of CSMA/CA running on a sectored AP, and provides a sector table for simple AP routing functions. The enhanced functionality for the AP introduces several tradeoffs, such as increased complexity and cost, that we believe are far out-weighed by the dramatic increases in throughput and efficiency, translating into a system that can handle an increased customer base with higher customer satisfaction.

7. REFERENCES

1. B.P. Crow, I. Widjaja, L.G. Kim, P.T. Sakai, "IEEE 802.11: Wireless Local Area Networks," *IEEE communications Magazine*, vol. 35, no. 9, pp. 116 –126, September 1997.
2. "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," Draft Standard, IEEE 802.11, P802.11/D1, 1997.
3. J.-L.C.Wu, H.-H. Liu, and Y.-J. Lung, "An adaptive multirate IEEE 802.11 wireless LAN," in Proc of *International Conference on Information Networking*, pp. 411-418, 2001.
4. Eun-Seon Cho, Go-Whan Jin, Cheol-Hye Cho, "Comparisons of mobility models in cellular systems," Vehicular Technology Conference, pp. 19-22, September 1999.
5. F. Shad, T. Todd, V. Kezys, and J. Litva, "Indoor SDMA Capacity Using a Smart Antenna Basestation," in *Proc. of IEEE ICUPC.97*, pp. 868-872, 1997.
6. T.S. Rappaport, *Wireless Communications: Principles and Practice*, 2nd edition, Prentice Hall, Upper Saddle River, New Jersey, 2002.
7. Y.-B. Ko, V. Shankarkumar and N.H. Vaidya, "Medium Access Control Protocols Using Directional Antennas in Ad Hoc Networks", *IEEE INFOCOM*, pp. 13-21, March, 2000.
8. K. Pahlavan, P. Krishnamurthy, *Principles of Wireless Networks*, 1st edition, Pearson Education, Singapore, 2002.
9. Bandyopadhyay, S.; Pal, M.N.; Saha, D.; Ueda, T.; Hasuike, K.; Pal, R, "Improving system performance of ad hoc wireless network with directional antenna", *IEEE International Conference on Communications*,Volume 2, May 2003.

SERVICE-DRIVEN GROUP MANAGEMENT FOR MOBILE P2P SERVICES

A. Liotta[1], M. Ballette[1], L. Lin[1], M. Gasparoni[2], P. Brick[2], N. Papadoglou[2]
[1]ESE Department, University of Essex, Wivenhoe Park, Colchester CO4 3SQ, United Kingdom. [2]Vodafone Group Services Ltd. Vodafone House, The Connection, Newbury Berkshire RG14 2FN, United Kingdom

Abstract: Given the considerable trend towards multi-party, peer-to-peer (P2P) communication, many are looking at the P2P computing paradigm as the means to extend the capability and scalability of Internet-based services. Current P2P frameworks are, however, largely incompatible with each other and do not address all the requirements of mobile computing. Here we propose a novel solution to peer group management which is 'autonomic', 'mobile friendly' and 'service driven'. We illustrate how our approach facilitates mobile P2P services by managing peer groups based on service semantics and resource availability. Our system allows efficient 'deep' search of user personal content stored in thin mobile terminals. Results are based on an experimental prototype and are demonstrated by a simple proof-of-concept mobile service.

Keywords: Mobility aware P2P; Mobile P2P applications; P2P Group Management.

1. INTRODUCTION

Mobile services over all-IP networks are revolutionizing the way people communicate, enhancing their ability to engage in virtual collaborations and share content. The classic client-server (C-S) interaction paradigm is still the backbone of Internet services. However, as these are increasingly based on multi-party, Peer-to-Peer (P2P) communication, P2P computing is gaining momentum as one of the fundamental building blocks of modern services.

Mobile P2P (MP2P) services bring the benefits of P2P computing (e.g., increased scalability, robustness, seamless resource pooling) to the mobile user. Despite their appeal, MP2P services are difficult to provide today since existing P2P frameworks are heavyweight and not geared towards mobility. On the other hand, mobile terminals are thin and involve intermittent connectivity that is not typical in conventional P2P computing.

Beyond the plethora of Internet services, mobile services represent an unprecedented revenue-raising opportunity for mobile network operators and value-added service providers. In particular, operators hold a unique position in the service-provisioning marketplace. They can rely on standardized service-centric frameworks (such as IMS[1]) and on a pre-existing range of subscribers. They have relevant know-how and are in the best position to add new dimensions to P2P services, namely security, authentication and quality of service.

Taking the operator's perspective, we present here a novel approach to deploying MP2P services which is 'autonomic', 'mobile friendly' and 'service driven'. In particular, we address the problem of providing effective group management in such a way which allows thin mobile terminals to dynamically join P2P services, share content and initiate virtual P2P communities.

Our approach is service-driven – peer groups or clusters are created on the basis of service semantics. We illustrate this new approach via a proof-of-concept P2P property selling application (Section 2), where peers are clustered depending on their postal code of interest. A completely different grouping strategy would be used, for instance, in a MP2P gaming service, where users may want to use group-joining strategies based on game name, user ability, location and so forth. Group structuring/organization is fundamental to P2P services. It affects the way P2P resources (including services) are, first, advertised and published and, then, discovered. The way adverts are organized, eventually affects the scalability of the MP2P service so we argue that the operator or service provider should directly affect that. This is why in our system grouping arises directly from service requirements.

Upon capturing the most relevant P2P systems (Section 3), we explain why our approach is mobility-aware (Section 4). Finally, in Section 5, we present the evaluation of our approach. Having realized a core set of MP2P protocols (publish, discover etc) and a MP2P service incurring a memory consumption of just 600Kbytes, we have proved the viability of our ideas.

2. MP2P SERVICE DEPLOYMENT SCENARIO

The application and use case scenario depicted in Fig. 1 have been developed for the purpose of illustrating the potential of mobile P2P services, demonstrating their viability in relation to state-of-the-art technologies, and present the service-driven group management approach proposed in Section 4. We are assuming a scenario in which mobile services are offered via a mobile network operator *e.g.* through the IP Multimedia System (IMS)[1], which supports the initial service discovery, registration, authentication and so forth. A typical sequence of interactions is depicted in Fig. 1. A user instantiates the service with the purpose of advertising a property. In addition to typical information (address, type of property, price etc) the user may use the terminal camera to grab/store pictures or video clips of the property. After entering the property data, an XML advert is generated (Fig. 1 step 1). This will be subsequently used during the semantic discovery phase (see Sect.4). Having stored the content and its corresponding XML advert, the user uses the IMS framework to authenticate itself and register to the property-selling service in the role of a "seller" (2).

A buyer joining the server will go through the usual operator's authentication process (3, 4), followed by a discovery phase. P2P publish and discovery are described in Section 4, while Fig. 1 shows that, once the buyer discovers the seller, any further communication between them is in P2P mode – i.e. the operator is bypassed for efficiency reasons (5). Being part of a P2P system, the buyer now transparently acts also as a server for any content it has downloaded from the seller (6). Other users may, then, instantiate the service either as buyers, sellers or both (steps 7-10).

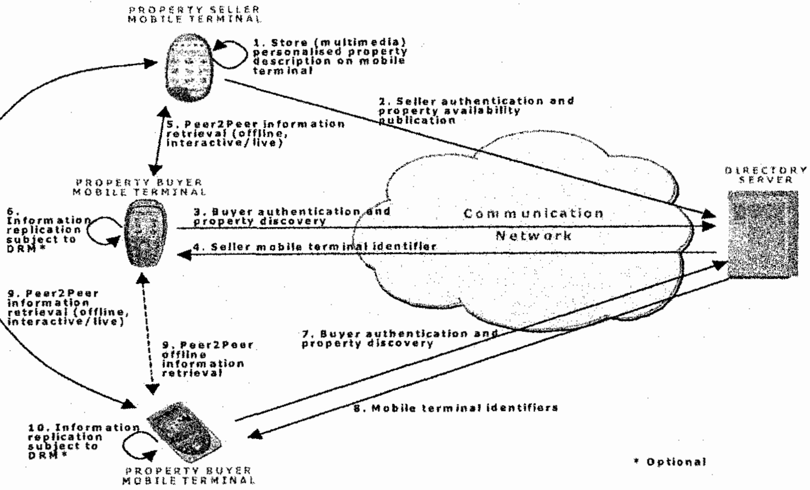

PROPERTY SELLER MOBILE TERMINAL
1. Store (multimedia) personalised property description on mobile terminal
2. Seller authentication and property availability publication
5. Peer2Peer information retrieval (offline, interactive/live)
PROPERTY BUYER MOBILE TERMINAL
DIRECTORY SERVER
6. Information replication subject to DRM*
3. Buyer authentication and property discovery
Communication Network
4. Seller mobile terminal identifier
9. Peer2Peer information retrieval (offline, interactive/live)
9. Peer2Peer offline information retrieval
7. Buyer authentication and property discovery
8. Mobile terminal identifiers
10. Information replication subject to DRM*
PROPERTY BUYER MOBILE TERMINAL
* Optional

Figure 1. A personalized MP2P property selling application.

3. EXISTING P2P SYSTEMS

The Peer-to-Peer (P2P) computational paradigm[2] is in essence an alternative to the Client-Server (C-S) model. We can identify four main approaches to P2P computing. *Hybrid P2P systems* such as Napster[3], Jabber[4] and ICQ[5] represent the first generation of P2P systems. They use a centralized indexing server to facilitate the interaction between peers – but this introduces, again, a single point of failure and a bottleneck.

In a *pure unstructured P2P system* all nodes communicate and find each other directly through a P2P overlay *with no central server intervention.*

This solution is highly *scalable* and *reliable* since a failure in one or more nodes does not affect the communication between the remaining nodes. Examples are Gnutella[6] and Freenet[7]. The inherent drawbacks are the limited search horizon and the large network overhead (queries are propagated via flooding). Hence, unstructured systems have been superseded by *structured P2P systems* which are based on *Distributed Hash Tables* (DHT)[8]. The latter systematically reduces search complexity - examples are Chord[9], Pastry[10], and Tapestry[11]. What limits the applicability of DHT systems to the area of mobile services is their simplicity in terms of searching.

Super-Peer networks such as JXTA[12,13] consist on some special nodes that operate as a server to a set of clients and as an equal in a network of super-peers. These systems improve manageability without compromising *scalability* and *fault tolerance.* They combine the efficiency of centralized search with the autonomy, load balancing and robustness provided by distributed search. The key benefits exploited also in our system are the reduced search time and the limited signaling overhead. If we consider M Super Peers in charge of N peers (with $N>>M$), a search will take O(N) in a pure unstructured P2P overlay and O(M) in a Super Peer-based overlay.

4. GROUP MANAGEMENT IN PEERMOB

The work presented herein is part of the PeerMob project, whose aim is to build a service framework facilitating the deployment of mobile P2P services by network operators. Given the scope of this article, below we focus on the group management aspects of PeerMob.

4.1 Semantic Clustering for Service-Driven Grouping

PeerMob adopts a Super Peer, n-tier hybrid architecture where the number of hierarchical levels and the structure of the P2P overlay are determined by service semantics. We illustrate this novel approach by means of an example, using the application of Section 2. For the sake of simplicity, Fig. 2 depicts a possible solution based on a 3-level hierarchy. The main components are:

- *PeerMob centralized server*: holds the service schema which ultimately influences they way peers are dynamically grouped, acting as a meta-information repository of Super Peer (SP) identifiers;
- *Super Peers*: realize typical cluster head functionality, acting as information hubs (for discovery requests across groups), and store the meta-adverts corresponding to the whole content stored in their group peers.

- *Simple Peers (or peers)*: Store the actual data content (e.g. pictures, video clips, property description) to be shared with other peers.

Looking at the requirements and specification of the P2P application, the service developer designs the service schema. We have used XML to facilitate semantic data processing. For efficiency, our prototype makes use of a relational database. Fig.3 illustrates a snippet of the server-side meta-information w.r.t. the application of Section 2 and the scenario depicted in Fig.2. Fig.3A, illustrates the grouping based on postal code; Fig.3B, shows the three SP identifiers; and Fig.3C identifies the allocation of SPs to Groups (one SP per group in this simple case).

Service semantics influences also the structure of the meta-information maintained by Super Peers. Our SPs have a lightweight relational database analogous to the one present at the server side. However, while the server-side meta-information relates groups with SPs, the SP meta-information relates SPs with simple peers.

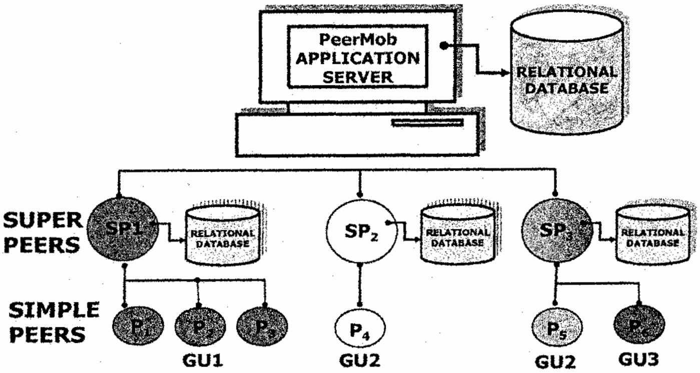


Figure 2. The PeerMob system in relation to the application of Section 2.

4.2 Mobility-aware Group Management

All meta-information is dynamically updated as new groups appear and new users join/disjoin those groups. Clearly, when the size of a group grows (for instance as a result of an increased interest in the properties of a particular area), the number of SPs per group should grow too. Since we are assuming a mobile P2P scenario, relatively thin terminals may act as SPs; so another important element of group management is the run-time level of congestion of SPs. Finally, since we are assuming that any peer (including SPs) may be mobile, the system should keep track of who is connected and react appropriately when peers lose connectivity. In particular, since SPs store the adverts relating to the data content shared by simple peers, it is essential to maintain replicas of those adverts. Similarly, the actual content stored by simple peers should be transparently replicated to increase availability and tolerance to loss of connectivity.

We have addressed the aforementioned requirements via an autonomic, self-management approach. This is based on periodic messaging that is confined between adjacent hierarchical levels. Going back to the example of Fig.2, peers periodically report their presence directly to their SP (or SPs to cater for data replication). In turn, SPs report their presence and level of resource utilization (the 'capacity' field in Fig.3B) directly to the centralized server.

In this way, when a simple peer loses connectivity from its group, the relevant SP can trigger a P2P fault-tolerance mechanism. For instance, if the disconnected peer contains data which is requested by some other peer, the SP dynamically redirects any request meant for the disconnected peer to the peer containing its replicas. Loss of connectivity of a SP is handled in a similar way but involves the intervention of the centralized server.

Group size is managed on the bases of the 'capacity' information relayed by SPs to the centralized server and of the 'peer to SP' ratio. So if a SP is hit by 'too many' requests, and as soon as its run-time resources go below a certain threshold, a new SP election mechanism is triggered under the control of the server. A similar reaction is triggered as the number of peers per group increases.

```
<PeerMob>
  <Groups>
    <group name="GU1
    <group name="GU2
    <group name="GU3
  </Groups>
  <SuperPeers>
  </SuperPeers>
  <Relations>
  </Relations>
</PeerMob>
```

A

```
<Groups>
</Groups>
<SuperPeers>
  <superPeer peerID="PeerMob.1109261307758">
    <peerName>sp1</peerName>
    <peerIP>213.205.32.10</peerIP>
    <peerPort>80</peerPort>
    <capacity>354</capacity>
    <advCachedNum>124</advCachedNum>
    <refreshTime>1109292348647</refreshTime>
  </superPeer>
  <superPeer peerID="PeerMob.1109261304556">
    <peerName>sp2</peerName>
    <peerIP>245.168.10.65</peerIP>
    <peerPort>80</peerPort>
    <capacity>30</capacity>
    <advCachedNum>20</advCachedNum>
    <refreshTime>1109292345698 5</refreshTime>
  </superPeer>
  <superPeer peerID="PeerMob.1109261307951">
    <peerName>sp3</peerName>
    <peerIP>122.205.32.10</peerIP>
    <peerPort>80</peerPort>
    <capacity>367</capacity>
    <advCachedNum>24</advCachedNum>
    <refreshTime>1109292366322 1</refreshTime>
  </superPeer>
</SuperPeers>
<Relations>
```

B

```
<PeerMob>
  <Groups>
  </Groups>
  <SuperPeers>
  </SuperPeers>
  <Relations>
    <relation relationID="PeerMob.1109261307951.GU1">
      <peerName>sp1</peerName>
      <groupName>GU1</groupName>
    </relation>
    <relation relationID="PeerMob.1109261307951.GU1">
      <peerName>sp2</peerName>
      <groupName>GU2</groupName>
    </relation>
    <relation relationID="PeerMob.1109261307951.GU1">
      <peerName>sp3</peerName>
      <groupName>GU2</groupName>
      <groupName>GU3</groupName>
    </relation>
  </Relations>
</PeerMob>
```

C

Figure 3. Server-side XML schema for the property selling application of Section 2.

4.3 Semantic Publishing and Discovery

P2P services critically rely on two functions, publish and discovery. Publishing, allows a peer to advertise its content, resources or services in such a way as to make them shareable with other peers. Discovery, in turn, allows a peer to find resources of interest. PeerMob adopts a semantic approach to publish and discovery. The first issue for a seller is to determine the group where its advert should be propagated. In our case, semantic indexing is achieved by mapping the seller's advert (in XML) with the server DB information. Given the semantics of this particular service, the server returns SPs which store adverts w.r.t. the relevant postal code. Multiple SPs are returned in order to allow the replication of adverts. At this point, the peer joins the relevant peer group and publishes the advert in SP_1 and SP_2. A similar semantic-based procedure is followed by the buyer. In this case, the initial task is to determine the relevant group that holds adverts regarding the geographic area of interest. Upon joining a group, the buyer can query the group's SP, issuing an XML query which is easily matched against the SPs list of adverts. The SP finally returns the list of sellers to the requester who can now engage in P2P communication with the buyers.

5. SYSTEM EVALUATION

We have realized a first prototype of the MP2P core along with the application of Section 2, proving the viability of the MP2P concept involving thin mobile terminals – the total footprint is only 600Kbyte. In the remainder we present a significant sample of results aiming at assessing response time, overheads, and scalability.

5.1 Super Peer Response Time

Super Peers store adverts and handle queries coming from group members or from other SPs. We are assuming a mobile environment in which wireless enabled PDAs can act as SP. We assess here the ability of such thin devices to act as SPs. The experimental set-up is illustrated in Fig. 4. The PDA is an HP IPAQ 5550 with a 400 MHz Intel XScal PXA255 processor, 128 Mbytes RAM, 48Mbytes ROM and integrated WiFi (802.11b). Multiple simple peers run on a single laptop. The PDA stores 1000 XML adverts in its cache and is hit by an increasing query rate ranging between 2 and 32 requests per second. Fig. 5 illustrates the results, including average values and the 95% confidence bands related to 100 repetitions. The important result is that response time does not increase dramatically, proving that even thin terminals can be effective SPs. Scalability is achieved thanks to the distributed search mechanism offered by P2P.

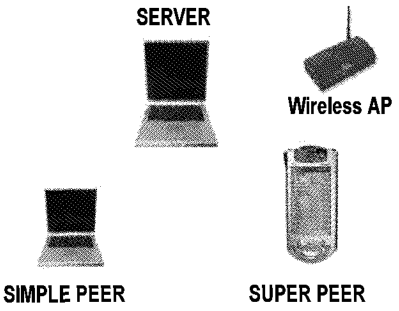


Figure 4. Experimental set-up.

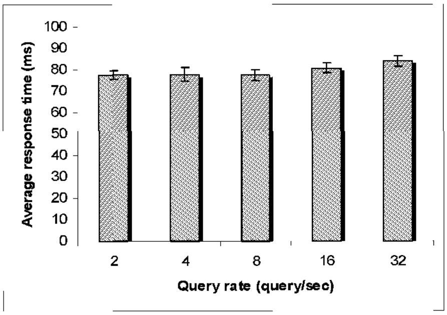


Figure 5. Super Peer response time.

5.2 System signaling

Like for any other distributed system, P2P systems offer advantages in terms of response time and scalability which are paid at the expenses of other forms of overheads. In PeerMob there is periodic messaging between adjacent hierarchical levels (Fig.2). The advantage of our approach is that messages generated by simple peers are confined within their group (i.e. do not propagate up to the server). There is, however, messaging between SPs and server – this is needed for effective group management (Section 4). Messaging around the server is the main determinant of bottlenecks – we report here our findings based on mathematical modeling (for brevity, the model description has been omitted).

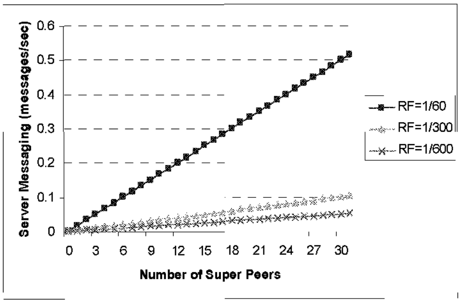


Figure 6. Server messaging Vs. SPs.

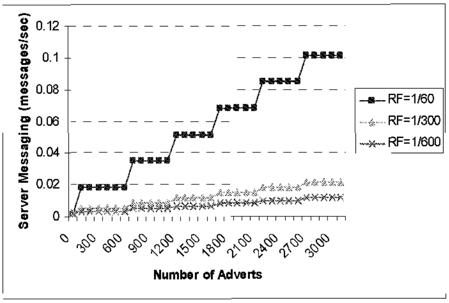


Figure 7. Server messaging Vs. adverts.

The linear behavior of Fig.6 was expected since the size of messages does not change with other system parameters, whereas message rate is determined by the self-management system. As the system becomes more dynamic, the message refresh rate (RF) has to increase and so will the slope of the line.

Fig.7 captures what happens as we increase the number of adverts. In this case the maximum SP capacity was set to 500 adverts. This means that while the number of adverts increases linearly, the number of SP increases only when its maximum capacity is reached (every 500 adverts) – hence the step-like shape (message rate increases only with the number of SPs).

Being resilient to peer failure, PeerMob adopts a messaging rate which varies with the Peer Failure Frequency (PFF). Fig. 8 captures the influence of this parameter.

5.3 Search Failure Rate

Fig.9 captures the system resilience to SP failure. The result highlights a very attractive property which is typical of P2P systems, i.e. their ability to increase resource availability via data replication mechanisms. In our study we emphasize the effect of SP failure probability since this is an important factor in mobility-aware P2P systems. We can see that our system can compensate to a great extent SP failure by increasing data replication.

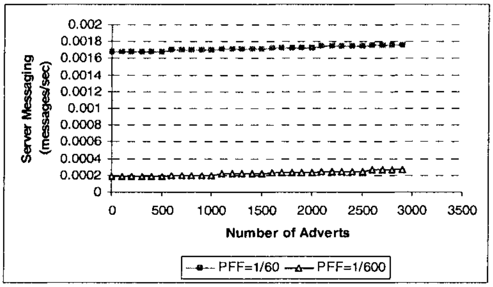


Figure 8. Effect of Peer Failure Frequency on server messaging.

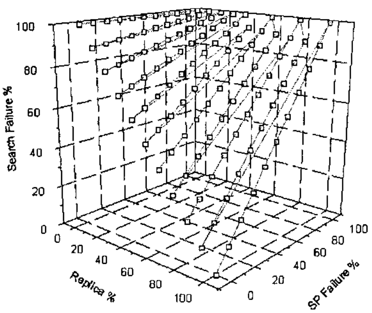


Figure 9. Resilience to SP failure.

6. CONCLUDING REMARKS

Behind the scenes of this article we are addressing the following questions: is it possible to satisfactorily bring the power of P2P to the mobile context? Whose role is it to provide the necessary level of management which can make this vision a reality? Can P2P-enabled mobile services spark a new generation of services?

The work carried out so far, indicates positive answers to the above questions and highlights the important role that operators may take.

Operators and service-providers already adopt a service-centric approach aimed at sparking a wealth of new mobile services. MP2P is certainly not the only enabler of revolutionary services but it can, indeed, play an important role if integrated with existing frameworks.

Our immediate plans are to integrate the PeerMob system in IMS and study the benefits that MP2P can bring into it, while also exploiting the standardized features of IMS in P2P. Security, authentication, and charging are certainly weak aspects of existing P2P frameworks that are also not geared towards mobility. A seamless integration of P2P computing with state-of-the-art service-centric frameworks such IMS seems the obvious step towards effective mobile P2P services.

ACKNOWLEDGEMENTS

The work described herein has carried out in the context of the PeerMob project sponsored by Vodafone Group Services Ltd., UK.

REFERENCES

1. M. Poikselka, et al., The IMS: IP Multimedia Concepts and Services in the Mobile Domain. John Wiley & Sons, 2004.
2. M.P. Singh, Peering at Peer-to-Peer Computing. IEEE Internet Comp., Vol. 5(1), 2001.
3. http://www.napster.com
4. http://www.jabber.org/
5. http://www.icq.com/
6. http://www.gnutella.com/
7. http://freenet.sourceforge.net/
8. Zhiyong Xu, Rui Min, Yiming Hu, Reducing Maintenance Overhead in DHT Based Peer-to-Peer Algorithms. Proc. of P2P'03.
9. I. Stoica, et al., Chord: A scalable peer-to-peer lookup service for internet applications. Proc. ACM SIGCOMM 2001.
10. A. Rowstron, P. Druschel, Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany, November, 2001.
11. B. Zhao, J. Kubiatowicz, A. Joseph, *Tapestry: An Infrastructure for Fault-Resilient Wide-Area Location and Routing*. Report UCB/CSD-01-1141, U. C. Berkeley, 2001.
12. http://www.jxta.org/
13. S.M. Botros et al., Search in JXTA and Other Distributed Networks. Peer-to-Peer Computing, 2001.

TOWARDS PROGRAMMABLE CONTEXT-AWARE VOICE SERVICES

Kerry Jean, Nikolaos Vardalachos, Alex Galis
Department of Electronic Engineering, University College London,
Torrington Place, London, WC1E 7JE, U.K.; Tel: +44-20-7679-5752
{kjean, nvardala, agalis@ee.ucl.ac.uk

Abstract. Programmable context-ware services use context information and programmable networks technology in the provision of easily customised and personalised services, which can respond appropriately to changes in their environment. This paper presents one such service, which is used to enable the provision of VoIP services in crisis situations. This service, the context-aware VoIP (CaVoIP) service is built upon the CONTEXT platform, an innovative middleware designed for the creation, deployment and management of context-aware services. The platform consists of a programmable layer, a context-aware service engine and a policy-based service layer. The voice services in the CaVoIP service are provided by a session initiation protocol (SIP) platform called Siptrex. The result is an easily customised, flexible and scalable context-ware service, which suppresses, non-essential traffic during crisis situations allowing greater bandwidth for essential traffic.

Keywords: programmable services, programmable networks, context, and context-aware services

1 Introduction

Context consists of the implicit and explicit information of an entity, be it an application, network or service, which can be used to characterise it. This context information can be used to enhance a service or application, personalising it, enriching it or making it more responsive to changes in its environment or situation [1]. When a service makes use of its context, it becomes context-aware. Context-awareness is characterised by the ability of a service or system to react to its changing environment. Context-aware services can be developed through the creation of a context infrastructure on top of a programmable network [3][6]. This paper presents a programmable context-aware service designed to efficiently use network resources to cope with the huge increase in voice traffic during crisis situations.

This research was carried out as part of the an EU funded project called CONTEXT [3][4], which created the CONTEXT platform. This platform made use of programmable networks and policy-based service management (PBSM) for the efficient creation, deployment and management of context-aware services. The CONTEXT project created a context-aware VoIP (CaVoIP) service where context and programmable networks are used to enhance a VoIP service enabling it to react appropriately to crisis situations such as terrorist attacks or major disasters. During a crisis there is a huge strain on a network due to the great increase in voice traffic, both essential from the emergency services, doctors, hospitals) and non-essential (from general public). Without proper management, there is less bandwidth available for essential traffic (resulting in a degradation of the quality of the service. The aim of the CaVoIP service is to suppress non-essential traffic and only allow traffic from priority users to be carried on the network. This service would monitor for different crisis conditions and enable the network to respond appropriately. In the CaVoIP service a session initiation protocol (SIP) [2] VoIP platform called Siptrex [5] is used to provide voice services.

The next section, the background, provides an overview of the technologies used to create the service namely programmable networks, context and context-awareness. Then the service specifications, realisation, components and interactions of the CaVoIP service are detailed. A service scenario and evaluation is then presented. This paper ends with the conclusions.

2 Background

2.1 Context and Context-awareness

Context is defined as any information that can be used to characterise the situation of an entity, where an entity can be a person, place, physical or computational object. Typical examples of context information are: (1) user location information (e.g. outdoors/indoors, street, city, etc.); (2) social context (e.g. role –student/staff/faculty; wife; boss; colleague, etc.); (3) personal preferences (e.g. food preferences, favourite sports, etc.); (4) user's behaviour (e.g. task, habits); (5) device and network characteristics (e.g. network elements, bandwidth).

A system is context-aware if it uses context to provide relevant information and/or services to the user. The user here can be a human end user or another application, service or system. Context-awareness enables a new class of computing and network services [3]. These services can be easily personalised to help users find nearby services or devices, decide the best devices to use, receive messages in the most useful and least intrusive manner, and can enable systems to react appropriately to certain situations etc. As the CaVoIP service is context-aware, it uses network information to respond appropriately to a crisis and can be easily personalised.

2.2 Programmable/Active Networks

Traditional networks passively transport data packets from one host to another via routers. Each node performs only the processing necessary to forward packets towards their destination. In programmable or active networks, the packets contain code, which can be executed by active routers. Active or programmable routers are network nodes that execute the code contained in an active packet [10]. This code can be used to make the network nodes more intelligent and programmable. Active network architectures enable a massive increase in the complexity and customisation of the computation that is performed within the network [14].

There exist two main approaches to realise active networks: *programmable nodes* and *encapsulation.* In the first approach, the user injects programs into the programmable node separately from the actual packet using existing network packet formats and providing a discrete mechanism for downloading programs to the active nodes. The program is executed when data packets associated with it arrive at the node [10] [13].

In contrast, in the *capsule* approach, a program is integrated into every packet. Encapsulation replaces existing packet structures with programs that are encapsulated within the transmission frames. In this approach, the active node has a built-in mechanism to load the encapsulated code, an execution environment to execute the code and semi-permanent storage where capsules can retrieve or store information [13].

3 The Context-aware VoIP Service

3.1 CaVoIP Service Specification

The CaVoIP service is a programmable network service which uses context-awareness and PBSM to deal with crisis situations in VoIP networks. These situations cause an upsurge in network traffic. This upsurge has to be managed carefully to ensure that the quality of the essential traffic (from emergency services, police, hospitals, government etc) is not compromised. The solution is to terminate all non-essential traffic when a crisis occurs and from then on only allow essential traffic. The CaVoIP service is built on top of the CONTEXT platform, a programmable, policy-based network platform. The CONTEXT service creation subsystem creates the service specific code and policies for the CaVoIP service. The network is monitored for a crisis using context-aware service code. The PBSM along with the programmable network is used to deploy and manage the service. To fulfil these objectives, the service needs:

- A means of obtaining and analysing context information
- A means of interacting with the underlying programmable network
- A means of interacting with the PBSM used to manage the service.
- A means of accessing the Siptrex platform.

3.2 CaVoIP Service Realisation

The CaVoIP service works as follows. A context computational object (CCO) monitors the network, computes context from local information and publishes it to the Context broker. The Context broker is a programmable network interface which allows the CaVoIP service to interact with the programmable network and obtain context information. A service execution condition evaluator (SICE) subscribes to this information and determines when a crisis has occurred. A SICE monitors a particular condition to determine when a service should start. A crisis is defined as the moment when calls made to the emergency number (911) have passed a predefined threshold in a fixed period of time. When a crisis occurs the SICE informs the PBSM and certain execution policies are invoked resulting in the deployment of a service level object (SLO) to the nodes in the crisis area. A SLO is a programmable application that provides a context-aware service. This SLO begins executing by terminating all non-essential calls to and from the crisis area. It then takes over call admission control from the SIP servers and only allows privileged callers access to the network. When the crisis is over the SLO stops executing, relinquishes call admission control and the CaVoIP service reverts to its default state.

The CaVoIP service is realised through the service code (the SICE, SLO and CCO), the Siptrex and CONTEXT platforms and the SIP and Context brokers. The CONTEXT platform is used to create, deploy and manage context-aware services. The brokers serve as interfaces between the programmable network and the service code. They allow programmable entities access to the network and context information and enable them to perform network reconfigurations needed to implement services. The user agents and SIP (session initiation protocol) servers of the Siptrex platform were modified to interface with the SIP broker and the service code. The Siptrex and CONTEXT platforms, brokers and service code are all described below.

3.2.1 Context Platform

The CONTEXT project [4] designed and developed an innovative platform and middleware solution to efficiently provide context-aware services making use of programmable networks technology and PBSM.

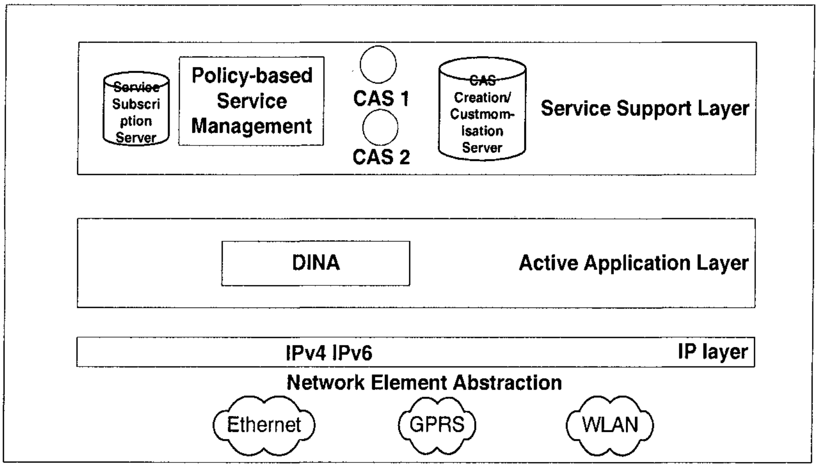

Service Subscri ption Server
Policy-based Service Management
CAS 1
CAS 2
CAS Creation/ Custmom-Isation Server
Service Support Layer
DINA
Active Application Layer
IPv4 IPv6
IP layer
Network Element Abstraction
Ethernet
GPRS
WLAN

Figure 1: CONTEXT framework architecture

Figure 1 illustrates the high level architecture of the CONTEXT platform [3]. It consists of a distributed service execution environment (EE) on top of a transport layer. The EE is composed of two layers, the service support layer (SSL) and the programmable application layer AAL. The features provided by these two layers are applied throughout the service creation, deployment, and operation phases of the service lifecycle.

The AAL provides the programmable network functionality to the higher layers. A programmable platform, DINA, was developed based on concepts used in ABLE [7][8][9]. DINA is a programmable middleware, which can be attached to different types of routers to make them programmable routers [10]. It enables the deployment of programmable services on network nodes. The APIs of the DINA platform allow the service code access to local, network, and context information, allowing it to perform actions (such as network level configurations) as needed. The ABLE platform was chosen, as it was lightweight, scalable and easily extensible. ABLE's extensibility and scalability results from the use of brokers, software components which enable access to all sorts of services and technologies e.g. SIP, context, QoS, GPRS etc. The creation of DINA was achieved by rewriting much of the ABLE C code in Java (to enable interoperability). During this process security was improved and support for context-aware services and IPv6 were provided.

The SSL consists of two main subsystems, the context-aware service (CAS) creation and customisation subsystem and the PBSM subsystem. The former enables context-aware service creation and customisation and also contains a service subscription server. The PBSM contains the policy management infrastructure. This consists of the policy manager, the code and policy repository as well as the code distributor and the code execution controller. Policies are used for service creation, deployment and operation.

The CONTEXT solution is characterised by three phases:

- *Service Creation and Customisation Phase:* In this phase, the behaviour of a context-aware service is defined based on the capabilities of the AAL. This behaviour is modelled as a set of policy rules in XML. Based on that model and the capabilities of the AAL, the service code and policies are then generated using a code generator. The generated service code and policies are then customised according to the customer's specifications.
- *Service Deployment Phase*: The code and policies are stored in code and policy repositories respectfully. According to the code distribution policies for the CaVoIP service, the customised code and policies are distributed throughout the network to the code storage and execution points (DINA nodes).

- *Service Operation Phase:* After the code has been distributed to the execution points, the service awaits triggers, as defined by the respective code execution policies, for code execution to begin. In the Ca-VoIP service, the trigger to start executing the service is when a crisis has occurred. The trigger or event is raised by a service invocation condition evaluator (SICE) when it detects the context conditions identifying a crisis.

3.2.2 DINA Components

DINA is a modular and scalable programmable network platform that enables the deployment, control, and management of programmable services over networks entities such as routers, WLAN access points, media gateways, and servers in IP-based networks. In addition, DINA provides interfaces (brokers) that can be used by the programmable services to retrieve information and perform configuration operations on local nodes. Figure 2 below presents the main DINA platform components.

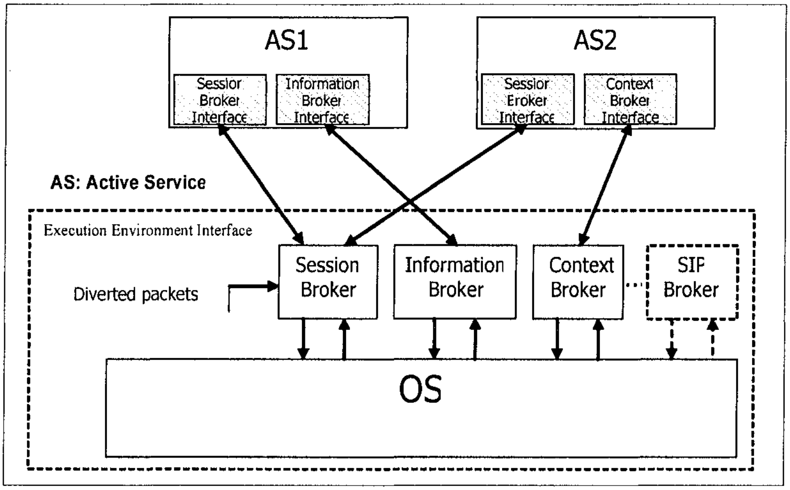


Figure 2: The DINA platform components

The DINA platform consists of two main components, the Session broker and the Diverter that run in parallel on each programmable node. The Session Broker is the core of the DINA platform. It receives and parses programmable packets, handles and manages existing services, and distributes programmable packets according to service requests. The Diverter receives the programmable packets captured by the programmable node and forwards them to the Session broker. In addition to these two components, other components, called brokers, run in parallel and provide enhanced services to the programmable sessions. These services extend the DINA platform allowing it to integrate with other technologies and services e.g. SIP, QoS, context etc. The following are two DINA components created for use in the CaVoIP service.

3.2.2.1 SIP Broker

The SIP broker is a programmable application, running on DINA, which interfaces between the AAL and the Siptrex platform used to implement the CaVoIP service. The broker provides to the programmable entities the ability to control and manage the SIP components such as proxy servers, and user agents. The SIP broker also provides the service code the ability to obtain information from the SIP server and to control aspects of call control during a crisis situation. The SIP broker performs three primary functions; provide information about the SIP servers, sessions and users; terminate voice sessions; and delegate call admission control to service level objects.

3.2.2.2 Context Broker

The Context broker implements the mechanism for enabling service code to access the necessary context information from the various context sources. It provides the methods that enable context producers to publish their context information and context consumers (service code) to retrieve it. In the CaVoIP service, it is used to store SIP, network and user context.

3.2.3 Siptrex System

The Siptrex system [5] is a versatile and extensible platform that provides VoIP services based on SIP. The Siptrex system software was developed in Java using the Java API for Integrated Networks (JAIN), the Java Media Framework (JMF) and a SIP parser from the National Institute of Science and Technology (NIST). The Siptrex platform was chosen to use in the CaVoIP service due to its simplicity, easy extensibility and interoperability.

The Siptrex system is of a client server design. The Siptrex clients (user agents) are SIP software phones (softphones) through which Siptrex users access the Siptrex services (through the Siptrex server) and are used to initiate and terminate SIP sessions. Siptrex service deployment is supported through the provision of a Siptrex API. The Siptrex platform components were modified to allow the SIP broker access to information from the Siptrex system enabling call admission control delegation and session termination.

3.2.4 CaVoIP Service Components

3.2.4.1 Service Invocation Condition Evaluator (SICE)

The SICE for the CaVoIP service is the SIP_SICE. It is configured to detect a crisis. It queries the Context broker to determine whether the number of SIP calls from a SIP domain has exceeded a certain threshold. When this threshold has been passed, a crisis is deemed to have occurred. The SIP_SICE then sends an event to the PBSM notifying it of an observed crisis.

3.2.4.2 Service Level Object (SLO)

The SLO for this service is the CH_Main. It is the key program in the CaVoIP service. It is deployed through the programmable network to the DINA nodes that are in the crisis area, as soon as the PBSM is informed of the crisis. Under a crisis situation the SLO takes over call admission control from the SIP server, allowing only privileged users access to the VoIP service. Non-privileged callers are blocked. Furthermore, the SLO terminates all non-essential calls when a crisis begins. It also fetches the user context (privileges) from the Context broker to determine whether they are privileged users.

3.2.4.3 Context Computational Object (CCO)

The CCO for this service is called the CH_Publisher. It collects SIP call information, relating to 911 calls, from the SIP broker. CH_Publisher compiles and publishes this information as context items to the Context broker.

3.3 CaVoIP Service Interactions

3.3.1 Service Creation and Deployment

The CaVoIP service is created using the CAS creation and customisation subsystem of the CONTEXT platform. The Siptrex and CONTEXT platforms are first installed. Then the Context broker, SIP broker and CH_Publisher are installed on the DINA nodes The CAS code generator generates the service code and policies for the CaVoIP service. The service code and policies are then customised to fit the requirements of the customer (e.g. the 112 versus the 911 emergency number, definition of a crisis situation etc.). The result is customised CH_Main and SIP_SICE, customised policies and customisation parameters for the CH_Publisher. The SIP_SICE along with the generated policies and customisation parameters is distributed to the code execution points (DINA nodes). This process is carried out by the code distributor and is controlled by a Distribute_Service_Code policy. Other policies control service code removal and revision.

3.3.2 Crisis Detection

The CH_Publisher registers to publish the 911 call context to the Context broker. The SIP_SICE is configured by the PBSM with specific policies, and subscribes to the context items associated with the statistics of the emergency number, and the crisis condition. At every reporting period the CH_Publisher asks the SIP broker running in the same domain for session statistics detailing the number of 911 calls. This context is then exported to the local Context broker. If a crisis condition is detected, the SIP_SICE is notified by the CH_Publisher through the Context Broker. Then the SIP_SICE creates a Start_CH_Main event and sends it to the PBSM. This event contains the address of the DINA node that hosts the SIP broker responsible for the domain in which the crisis has occurred. Upon receiving this event the PBSM distributes CH_Main to the specified DINA nodes. Figure 3, illustrates the interactions among the components of the CaVoIP service involved in crisis detection.

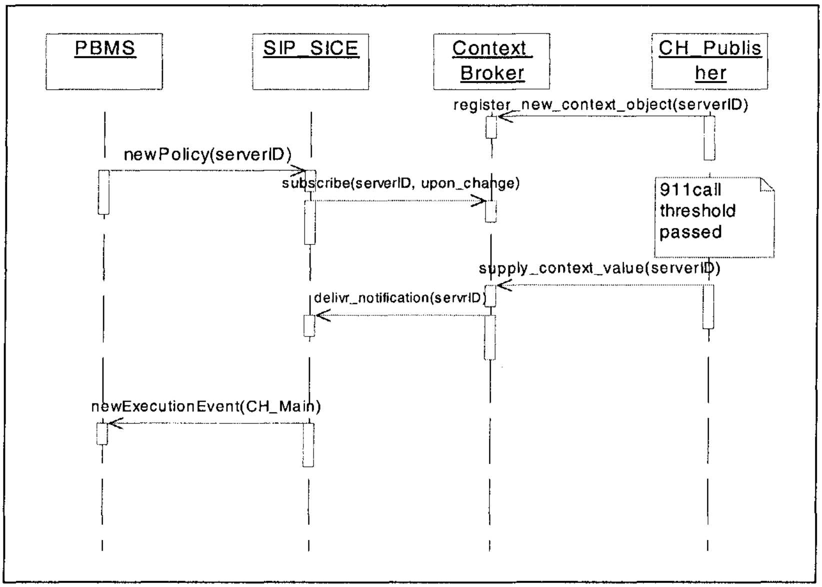

PBMS
SIP_SICE
Context Broker
CH_Publisher
register_new_context_object(serverID)
newPolicy(serverID)
subscribe(serverID, upon_change)
911call threshold passed
supply_context_value(serverID)
delivr_notification(servrID)
newExecutionEvent(CH_Main)

Figure 3: Sequence of interactions for the crisis detection stage

3.3.3 Service Execution

On arrival at the DINA node, CH_Main begins its execution by instructing the SIP broker to terminate all ongoing non-essential calls to and from the crisis domain. Then CH_Main is delegated call admission control. From this point on, all new sessions to be established must be authorised by the CH_Main. CH_Main checks the privileges of the callers and callees of all new call requests, by contacting the Context broker, allowing only privileged callers to make calls. When the crisis is over, CH_Main terminates and relinquishes call admission control. Figure 4 illustrates the interactions among components involved in the service execution stage.

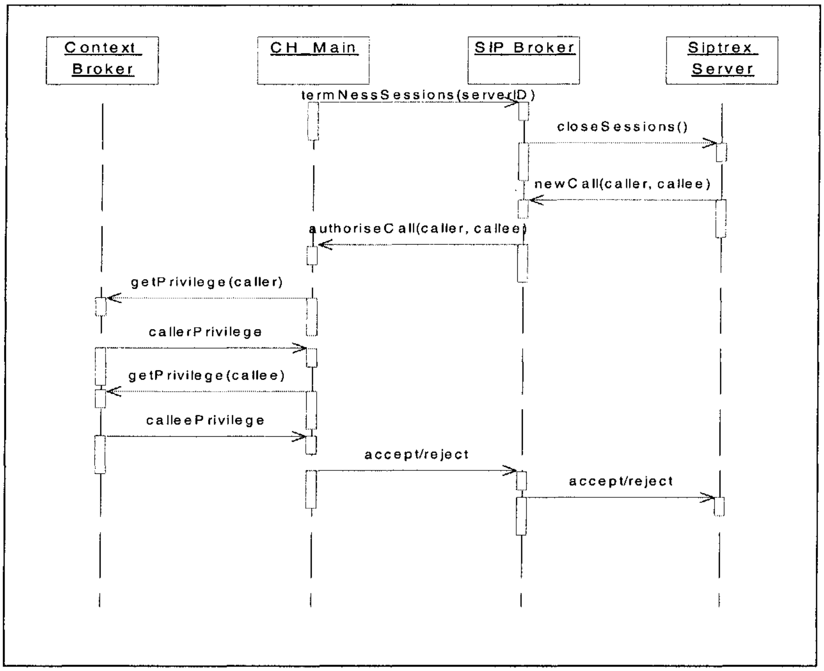


Figure 4: Sequence of interactions for the service execution stage

4 CaVoIP Service Evaluation

4.1 Service Scenario

A scenario can be envisaged where a terrorist bomb has exploded at a train station in a city. The scene is horrific and there are casualties. An eyewitness calls the emergency services. Many people gather at the site and there is a flurry of calls from them as well as other neighbourhood residents. The emergency services and police arrive and try to get the situation under control. An eyewitness calls the local newspaper to report the breaking news. When it has been established that a crisis is underway this call is terminated as the system terminates all the non-essential calls. Further call attempts fail and other low priority users cannot make calls. If a paramedic needs to call a doctor at the hospital for advice, the network accepts the call, due to his privileged status.

4.2 Service Testing

The aim of this test was to prove that the CaVoIP service works as envisaged during the design process. The test would be considered successful if as soon as a crisis occurs all the non-essential calls are dropped and from then on only essential calls are allowed. Figure 5 illustrates the testbed used for the CaVoIP service.

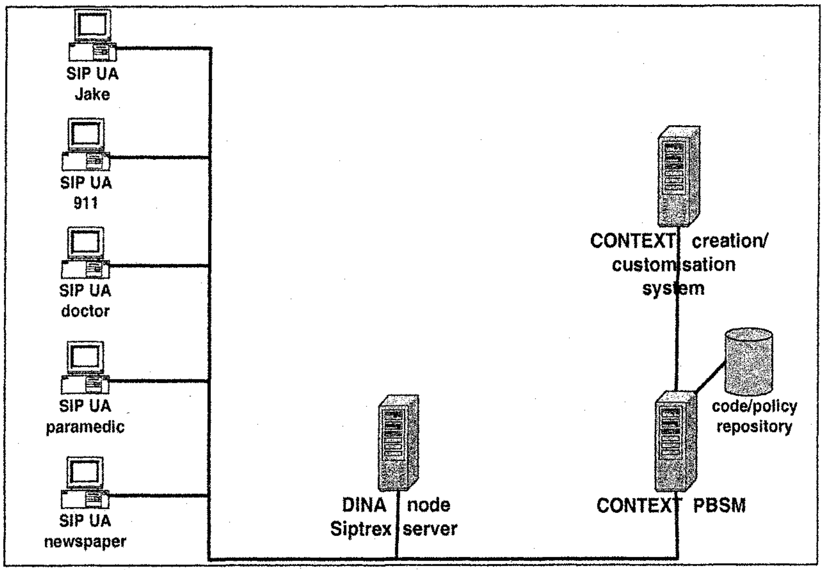


Figure 5: Testbed for CaVoIP service

The components of the CaVoIP service testbed are described below:

- One Siptrex server hosted on a Linux machine.
- One DINA node on the same Linux host as the Siptrex server. It contains the DINA programmable platform with the SIP and Context brokers as well as the CaVoIP service components: CH_Main, SIP_SICE and CH_Publisher.
- The CONTEXT PBSM, which consists of the code execution controller, the code distributor, the policy engine and the code and policy repositories.
- The CONTEXT service creation and customisation subsystem.

- Several user agents acting as callers and callees, one for the eyewitness Jake, one for the 911 contact centre, one for the newspaper and one for the doctor.

4.2.1 Test Procedure

The tests were conducted with the SIP_SICE, CH_Publisher, SIP broker, Context broker, Siptrex server, DINA components and CONTEXT platform.

1. The user privileges for each of the user agents were introduced in the Context broker.
2. Seven completed calls were made to 911 in less than 5 seconds (this was the definition of a crisis) and then a call was made between Jake's user agent and the newspaper. The crisis was detected and CH_Main deployed and started to execute in the DINA node.
3. A call was then made between the paramedic's user agent and the doctor's user agent.
4. Then another call was made between Jake and the newspaper.
5. After a period of time, an event was raised indicating the end of the crisis. Then another call was made between Jake and the newspaper.

4.2.2 Test Results

As soon as the crisis was detected and the CH_Main SLO began to execute, the call between Jake and the newspaper was terminated (as Jake is not a privileged user). The next call between Jake and the newspaper was not authorised and did not go through. The call between the paramedic and the doctor was authorised and went through. This was because the paramedic and doctor are privileged users. After the crisis was over the call between Jake and the newspaper went through. In fact all calls in the system were then authorised. The service tests were all carried out successfully and hence the CaVoIP service works exactly as was designed within the limits evaluated in the above test.

4.3 Service Extensibility and Flexibility

The CaVoIP service is very flexible and extensible due to its context-awareness and use of policies. More complex privilege allocation and logic to deal with it can be introduced. This can be done by altering the service logic found in the SLO to allow several permutations of privilege and access control to be used. The user privilege could be allocated through a number scale. For instance, a paramedic could have a privilege of two, a doctor three while the chief of police has five. Ordinary users will have a privilege of one. The decision to authorise the call could then be made through a combination of the privileges of the caller and callee. This can be achieved just by changing the user context published to the Context broker and the call admission logic in CH_Main.

Changes to the service can be introduced during the service customisation phase. The definition of a crisis situation could be changed (e.g. 10 calls to 911 in 5 seconds or 20 calls to 911 in one minute) or a totally new crisis situation can be defined (e.g. 20 dropped calls in one minute). The context-awareness and policy-based infrastructure allows such easy customisation and flexibility. The policy-based nature allows easy extensibility of the service. Some users could be allowed video calling, others voice only calls, some short message service (SMS) calls while blocking others, all according to their privileges.

4.4 Service Scalability

As both the underlying platforms used to create the CaVoIP service, the Siptrex and CONTEXT platforms, are easily scalable then the CaVoIP service can also easily scalable. To allow scalability, much use is made of the modular design of the CaVoIP service and the policy-based infrastructure. Additional Siptrex servers can be accommodated by using a policy defined naming scheme both for the individual services and the context associated with the service. For instance if a Siptrex server is called SS001, all the context associated with it would be preceded with this tag. Different crisis conditions could be defined for each Siptrex domain. These parameters can all be defined during the service customisation phase.

5 Conclusions and Future Work

The CaVoIP service presented is a programmable context-aware service. It uses context, programmable network technology and PBSM to enable a network to deal with the huge increase in voice traffic during a crisis situation. It is easily customisable, personalisable and extensible due to the use of context and a policy-based infrastructure. The CaVoIP service could easily be developed further through more complex privilege allocations and more complex logic to deal with them.. Also the services provided by the CaVoIP service can be extended to provide video calls and instant messaging in which case it could be possible to define privileges that allow users to either have video calling, voice only calls or short message service (SMS) calls according to their user privileges

References

[1] Dey, A. and Abowd, G., "Towards a better understanding of context and context-awareness". Proceedings of Workshop on the What, Who, Where, When and How of Context-Awareness, affiliated with the 2000 ACM Conference on Human Factors in Computer Systems (CHI 2000), The Hague, Netherlands. April, 2000.

[2] Sinnreich, H., Johnston, A., "Internet Communications Using SIP," John Wiley & Sons, New York, 2001.

[3] CONTEXT Consortium, "Context project deliverable D2.2: CONTEXT Architecture: Solution for provisioning and delivery of context aware services" ed. UCL, 2004.

[4] CONTEXT project website, http://context.upc.es/index.htm.

[5] Siptrex website, www.siptrex.net.

[6] Sygkouna, I. et al., "Context-Aware Services Provisioning on Top of Active Technologies," IFIP 5th International Conference on Mobile Agents for Telecommunication Applications (MATA 2003), Marrakech, Morocco 8-10.10, 2003.

[7] Kornblum, Jessica. Raz, Danny, Shavitt, Yuval. "The Active Process Interaction with its Environment," Computer Networks, 36(1):21--34, June 2001.

[8] Raz, D. and Shavitt, Y., "Towards Efficient Distributed Network Management," Journal of Network and Systems Management, September 2001.

[9] ABLE: The Active Bell Labs Engine http://www.cs.bell-labs.com/who/ABLE/

[10] Denazis, S. G., Galis, A., "Open Programmable & Active Networks: A Synthesis Study" IEEE IN 2001 Conference, Boston, USA, 6- 9 May 2001, ISBN 0-7803-7047-3.

[11] Galis, A., Denazis, S., Brou, C., Klein, C. (eds), “Programmable Networks and Programmable Network Management,” ISBN 1-58053-745-6, Artech House, London April 2004.

[12] Ebling, M., Hunt, G. and Lei, H., “Issues of Context Services for Pervasive Computing,” Proceedings of Workshop on Middleware for Mobile Computing, Heidelberg, Germany, 2001.

[13] Tennenhouse, D. L. and Wetherall, D. J., “Towards an Active Network Architecture” Computer Communication Review, Vol. 26, No. 2, April 1996.

[14] Tennenhouse D., Smith J., "A survey of Active Network Research," IEEE Communications Magazine, January 1997.

CONTEXT-AWARE SECURITY POLICY AGENT FOR MOBILE INTERNET SERVICES[1]

George Yee and Larry Korba
Institute for Information Technology, National Research Council Canada, 1200 Montreal Road, Bldg. M-50, Ottawa, ON, Canada K1A 0R6;
{george.yee, larry.korba}@nrc-cnrc.gc.ca

Abstract: The recent proliferation of e-services on the Internet (e.g. e-commerce, e-health) and the increasing attacks on them by malicious individuals have highlighted the need for e-service security. E-services on the mobile Internet (mi-services) are no exception. However, for mi-services, the level and type of security may depend on the user's security preferences for the service, the power of the mobile platform, and the location of the mobile platform (we label these UPL). For example, if the user is traveling through a particularly dangerous area known for previous attacks, the security protection should be adjusted to use mechanisms that are resilient to these attacks. We propose the use of a security policy that allows for various security options commensurate with UPL, in conjunction with a context-aware security policy agent that notifies the service provider to activate new security appropriate to a change in UPL.

Keywords: context-aware; software agent; security policy; mobile Internet; services.

1. INTRODUCTION

Internet-based e-services for banking, shopping, learning, healthcare, and Government Online have been growing rapidly and are now spreading themselves within the mobile Internet (Ho and Kwok, 2003; Mallat et al, 2004). However, these services are subject to malicious attack in one form or another. This leads to concerns over their security (Josang and Sanderud, 2003; Ghosh and Swaminatha, 2001; Joshi et al, 2001).

In order for mobile Internet services (mi-services) to be successful, they must be secured from malicious individuals who continuously try to compromise them. An effective and flexible way of managing security for mi-services is to make use of security policies. A mi-service security policy is a specification of what security measures will be used to protect the mi-service from security attacks. It should be noted that a security policy by itself does not guarantee that its stated security measures will be put in place or be complied with. That is an area of policy compliance that is outside the scope of this paper.

A mi-service provider makes use of a security policy to specify the security measures that it will use to protect its mi-services. However, this security policy may not match up with the security needs of the mi-service, depending on the user's security preferences for the service, the computational power of the mobile platform, and the location of the mobile platform. For example, suppose the security measure for an e-learning application is user authentication by means of a password. This authentication approach is known to be insecure. A security-sensitive consumer such as, for example, a defense contractor, may wish to add biometric authentication for an e-course on advanced weapons research. In such a case, the defense contractor would not want to use the provider's mi-service that only has password authentication. As another example, suppose the security measure is access control. The provider's security policy may provide access to 5 features of a mi-service, whereas a particular consumer may need access to only 3 features. In this case, the consumer may be reluctant to make use of this provider's mi-service, especially if the consumer can find another provider that only offers the features needed and at a lower price. As a third example, suppose the security measure for a mobile banking application calls for encrypting the communication channel using AES (Advanced Encryption Standard). However, the user's cell phone has insufficient computing power to compute AES with reasonable performance. Again, the consumer would find it unsafe (or impossible) to use the mobile banking mi-service. As a final example, suppose there is an area of a large city that is notorious for man-in-the-middle attacks against mi-services. Mi-service consumers try to avoid this area but occasionally they have to traverse it in order to get to their destination. Unfortunately, the mi-service provider cannot target this particular area for more effective security against man-in-the-middle attacks so that once again, the service consumer is faced with a difficult situation.

As a solution to these issues, we propose the use of a context-aware security policy agent that would initiate the best available security measures for a mi-service depending on the user's security preferences for the mi-service, the computational power of the user's mobile platform, and the location of the user's mobile platform. We refer to this combination of user preferences, power, and location as UPL. Thus, referring to the examples above, the agent would trigger biometric authentication according to the user's preference, trigger access control for 3 features instead of 5, initiate a less computational resource intensive encryption algorithm (with acceptable loss in effectiveness), and invoke more aggressive defenses against man-in-the-middle attacks, according to the values of UPL. We further propose that the available best security alternatives be stated in a mi-service security policy that is negotiated and agreed between the mi-service consumer and the mi-service provider prior to using the service. Security policy negotiation is outside the scope of this paper but is described in Yee and Korba (2005).

In the literature, there are many papers related to security policies. Security policies have traditionally been used to specify security requirements for networks and distributed systems (Varadharajan, 1990). More recently, they have been applied to manage security for distributed multimedia services (Duflos, 2002) and for very large, dynamically changing groups of participants in, for example, joint command of armed forces for some time period (Dinsmore et al, 2000). In addition, Ventuneac et al (2003) describe a policy-based security framework for web-enabled applications, focusing on role-based security policies and mechanisms. None of these authors use security policies containing selectable alternatives as we do in this work.

We note here that our use of context-aware security policy agents for mi-services is a form of service personalization. A key difference between mi-services and stationary Internet services is that mi-services are more personal (Chae and Kim, 2003). Ho and Kwok (2003) state that mobile service personalization is sought after by service consumers. Therefore our proposal for the use of context-aware security agents as a form of personalization should be welcomed by mi-service consumers.

The remainder of this paper is organized as follows. Section 2 defines mi-services, derives requirements for security policies, and gives an example of a security policy with alternatives that can be used with our context-aware agents. Section 3 describes our context-aware security policy agents and how they are used. Section 4 presents a discussion on operational and implementation requirements for the agents. Finally, Section 5 gives our conclusions and areas for future research.

2. MI-SERVICES AND SECURITY POLICIES

2.1 Mi-Services

A mi-service for the purposes of this paper is an Internet service accessible using a mobile device such as a cell phone or wireless PDA. Figure 1 shows a network view of mi-services. In this figure, the mobile ISP (Internet Service Provider) provides mobile wireless access to the Internet. The mi-service provider provides the actual service.

The mi-service provider has a security policy that specifies what UPL alternative security measures it will use to secure its service(s). The consumer has security preferences for the UPL alternative security measures that will be implemented for the mi-service. In addition, the security policy implemented for the mi-service is transparent to the mobile ISP, i.e. the latter does not need to provide any kind of special support for implementation of the security policy, beyond what it normally provides for secure communication (the security policy is implemented at a higher architectural layer). This is important since involving the mobile ISP in the security policy would introduce further necessity for negotiation and agreements and possibly overload the mobile ISP in terms of processing requirements. Examples of current mi-services accessible via a wireless PDA are Amazon.com (online retailer), optionsxpress.com (online stockbroker), and WebMD.com (health information and technology solutions provider).

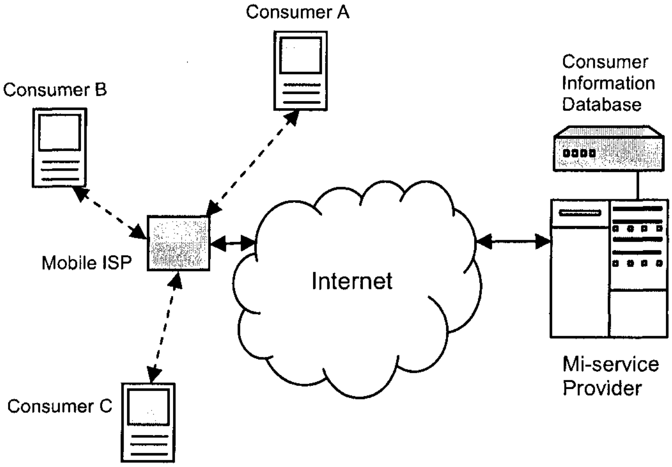


Figure 1. Network view of mi-services

2.2 Security policy requirements

Requirements for mi-services security policies address what security measures should be covered in a mi-service security policy. Since mi-services fall under the category of open systems, we begin by looking at requirements prescribed by ISO 7498-2, the reference model for security architectures by the International Organization for Standardization (International Organization for Standardization, n.d.). This standard identifies 5 main categories of security services: 1) Authentication, 2) Access Control, 3) Data Confidentiality, 4) Data Integrity, and 5) Non-repudiation.

The International Telecommunication Union Telecommunication Standardization Sector (ITU-T) provides Recommendation X.800, Security Architecture for OSI (Open Systems Interconnection) (International Telecommunication Union, n.d.) that lists the same 5 main categories of security services as above. We propose that these 5 categories of security services be covered in a mi-services security policy. We would add the following security services: 6) Secure Logging – of user transactions by the provider, 7) Certification – user or provider would use a certifying authority to certify credentials, 8) Malware Detection – user or provider would use some anti-malware software to detect and eliminate malware from their computing platforms, and 9) Application Monitoring – user mobile platform monitoring for licensed, verified, and permitted applications.

We thus have 9 security services that should be specified in a mi-service security policy. Figure 2 identifies where these security services are typically applied using a mi-service network view.

The above standards also list specific security services under the main security service categories. As an example, non-repudiation has the specific services (with the obvious meanings): "Non-repudiation, Origin" and "Non-repudiation, Destination". As well, security mechanisms (e.g. digital signature) are used to support security services, i.e. security policy requirements. We will employ specific security services and mechanisms to formulate our mi-services security policy.

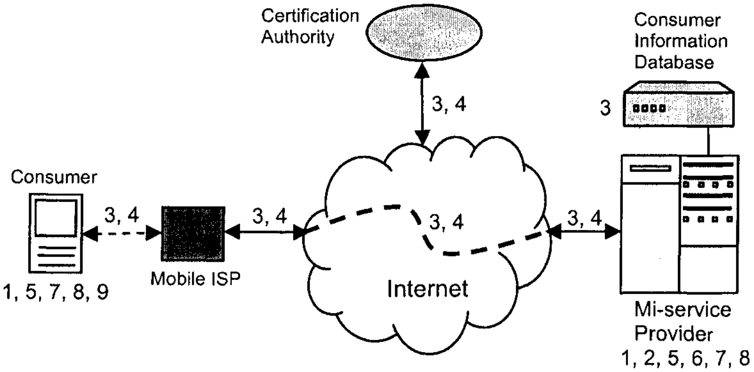


Figure 2. Application of security services (numbers correspond to security services in Section 2.2)

2.3 Mi-Service security policy

Based on the requirements of Section 2.2, and using example values and security mechanisms, we propose the example mi-service security policy shown in Table 1.

Table 1. Example mi-service security policy in schematic form

Policy Use: **My Service** ***Valid:*** **unlimited**	***Owner:*** **My Service Provider, Inc.**
CONSUMER PROVISIONS	**PROVIDER PROVISIONS**
Consumer Authentication *Implement:* yes (default) *P1: Mechanism:* password *P2: Mechanism:* V+F biometrics	**Provider Authentication** *Implement:* yes (default) *P1: Mechanism:* security token *P2: Mechanism:* digital signature
Consumer Non-Repudiation *Implement:* yes (default) *Mechanism:* digital signature	**Provider Non-Repudiation** *Implement:* yes (default) *Mechanism:* digital signature
Consumer Certification *Implement:* yes (default) *Mechanism:* certificate	**Provider Certification** *Implement:* yes (default) *Mechanism:* certificate
Consumer Malware Detect *Implement:* yes (default) *Mechanism:* Norton	**Provider Malware Detect** *Implement:* yes (default) *Mechanism:* Norton
Application Monitoring *Implement:* yes (default) *Mechanism:* IIT-ISG	**Data Store Confidentiality** *Implement:* yes (default) *Mechanism:* 3DES encrypt
	Communication Confidentiality *Implement:* yes (default) *P1: Mechanism:* SSL *P2: Mechanism:* VPN
	Communication Integrity *Implement:* yes (default) *Mechanism:* MD5 Hash
	Secure Logging *What:* order transactions *Mechanism:* 3DES encrypt *What:* user input *Mechanism:* 3DES encrypt
	Access Control *User Role:* Secretary *Resource:* scheduling module *Resource:* admin module *User Role:* President *Resource:* admin module *Resource:* salary module

In Table 1, the top shaded portion is the policy header. The header contains the following administrative fields: *policy use* identifies for which mi-service the policy is provided, *owner* identifies the name of the provider of the mi-service, and *valid* specifies the end date after which the policy is no longer valid. The *valid* field can also specify "initial" or "continuing" to indicate that the security policy is enforced only initially or continuously. The table also shows that some security services can have alternative mechanisms (e.g. consumer authentication using password or biometrics). These alternatives are prefixed by "Pn", where n is a number. The Pn are used by the context-aware security policy agent to select the associated mechanism for any particular invocation of the service. Further, secure logging and access control can have additional items (e.g. secure logging can log additional information and access control can have additional resources under each role). (Note: V+F biometrics refers to voice and fingerprint, IIT-ISG (Institute for Information Technology, Information Security Group) refers to a mechanism we are developing in our group.)

The security policy in Table 1 serves as the provider's security policy for a particular mi-service that the provider offers to consumers. It also reflects the consumer's security policy for the mi-service, since it contains provisions that the consumer agrees to follow. Upon locating the mi-service on the mobile Internet and prior to activating the service, the consumer examines the provider's security policy (Table 1) for the service comparing it to her own security preferences. If the consumer agrees with the provider's policy, the consumer can engage the mi-service. Otherwise, the consumer negotiates the security policy with the provider (Yee and Korba, 2005). If this negotiation is successful, the mi-service can start. Otherwise, the consumer needs to find a similar mi-service from a different provider (or find ways to match the security requirements of the present mi-service but it is probably easier to just find another mi-service), and repeat this process again. The security policy resulting from negotiation would be similar to Table 1, possibly with some security services not listed, and possibly with different alternative mechanisms or additional items for secure logging and access control.

3. CONTEXT-AWARE SECURITY POLICY AGENT

A context-aware security policy agent (CASPA) is an intelligent software agent that resides in a mobile device and is responsible for selecting security services and mechanisms from the provider's security policy for a particular mi-service, according to the values of UPL. The behaviour of a CASPA is described by the state machine in Figure 3, where the arrow labels are in the form "condition / action".

In Figure 3, the *Idle* state is exited once the service is ready to begin (i.e. the service has been found and the security policy agreed to between consumer and provider).

In the *Initialization* state, the CASPA accounts for the U and P of UPL (i.e. reflects the user's security preferences and the computational power of the device) by setting the options in the provider's security policy to implement appropriate security services and mechanisms (see Table 1). For example, suppose the consumer has several mobile devices that she uses with the same security policy, including a PDA and a less powerful cell phone. CASPA would set security services and mechanisms that both reflect the consumer's security preferences and be appropriate to the computing power of each device. It would be straight forward to program a CASPA to perform this task.

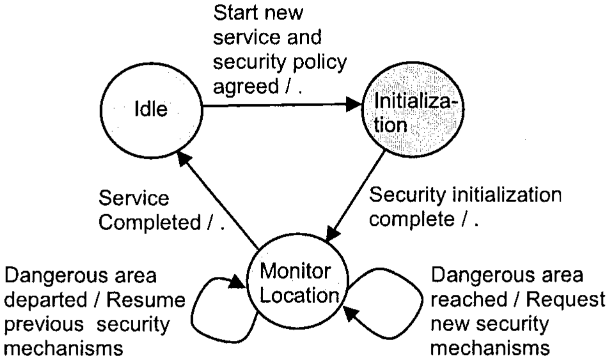

Start new service and security policy agreed / .
Idle
Initialization
Service Completed / .
Security initialization complete / .
Monitor Location
Dangerous area departed / Resume previous security mechanisms
Dangerous area reached / Request new security mechanisms

Figure 3. Behaviour of context-aware security policy agent

In the *Monitor Location* state, the agent is monitoring the device's location using GPS. Note that this location is only used by the CASPA and is not reported to either the mobile ISP or the provider of the service so that there should be no privacy concerns (more on this in Section 4). An alternative way of determining the consumer's location is the use of signaling analysis by the mobile ISP. However, the latter would then learn the consumer's location leading to privacy concerns. When a dangerous area (i.e. an area with a high number of attackers) is entered, the agent messages the service provider to initiate a more powerful security mechanism for communication to defend against the attackers (Section 4 discusses how this dangerous area can be known). Of course, this more powerful mechanism consumes more computing resources and should only be used when necessary. When the dangerous area is exited, the agent messages the provider that the normal security mechanism for communication may be resumed. The CASPA executes concurrently with the mi-service. However, the mi-service does not begin until the CASPA has completed the initialization.

The CASPA communicates with the provider during *Initialization* and *Monitor Location* using the following secure protocol:

1. $C \rightarrow P\colon Sig_C\ (M,\ nonce)$
2. $P \rightarrow C : Sig_P\ (nonce\text{-}1)$

where C is the consumer, P is the provider, Sig_C is the consumer's digital signature, Sig_P is the provider's digital signature, M is the message, and the nonce is used to prevent replay attacks and as a confirmation of receipt by the provider.

For *Initialization*, the message *M* has the form:

M = [INIT, security component 1, security component 2, ..., security component k],

where *security component j = security service j,* if this security service has no alternative security mechanisms, or *security component j = (security service j, mechanism idj),* if it has alternative mechanisms and *mechanism idj* is the mechanism the user wants.

For Monitor Location, upon entering the dangerous area, the message *M* has the form:

M = [NEW, (security service 1, mechanism id1), (security service 2, mechanism id2), ..., (security service m, mechanism idm)]

which sets the new mechanism of each security service that the consumer wants to implement for the dangerous area, for appropriate security services having alternative mechanisms. As we have alluded to above, in most cases the only security services of concern would be communication confidentiality and integrity. Upon exiting the dangerous area, the message M is: *M = [REVERT]* which tells the provider to revert to the previous mechanisms.

4. OPERATIONAL REQUIREMENTS AND DISCUSSION

The CASPA would need to know the user's security preferences, including the preferences for P and L from UPL, in order to formulate the messages *M.* These could be input via a UI for the CASPA. This information can be provided by the consumer once before any mi-services are used, and then verified with the agreed-to security policy for each service. The security preferences in *M* have to be realizable within the agreed-to security policy. In addition, the agreed-to security policies need to be expressed in a machine processable language such as XACML (eXtensible Access Control Markup Language) (OASIS, n.d.).

The provider needs to have software to receive the messages from the CASPA and apply them to the mi-service's security policy. This software could take the form of an agent as well, a counterpart to CASPA that acts on behalf of the provider.

In the *Monitor Location* state, an appropriate UI would be needed to interrupt the service temporarily while one or more security mechanisms are changed. This interruption occurs twice – once for entering the dangerous area and once for departing the dangerous area. Further, these changeovers need to occur quickly, in order not to annoy the user and to prevent any openings for attack. Dangerous areas may be determined as a result of feedback to a government website by users who have been attacked. The CASPA can periodically and automatically check this website for the latest dangerous areas.

The location obtained using GPS is only used by the CASPA and not reported to the providers which should not lead to privacy concerns. However, the dangerous areas are known to the service provider as well. The latter may infer the location of the consumer when the CASPA signals for higher security. We assume that this small breach of privacy is acceptable to the consumer in return for greater security, since the consumer's location may not be pinpointed exactly due to the possibility of more than one dangerous area and the fact that the consumer may enter a dangerous area at many different locations.

Our use of digital signatures and nonces implies that the mobile device needs at least the capability to process a digital signature and generate random numbers. In addition, there would need to be a key distribution technique, as well as the capability for the device to securely store a private key. However, these are minimal capabilities required to implement security services. Further, we require the mobile device to have a GPS capability, which is becoming more and more common. These requirements imply that the mobile device should probably have the computing power of a PDA. However, less powerful devices would be accommodated by the CASPA where possible.

We note that since the security policy is executed by the provider of the mi-service, the mi-service consumer can transparently use different mobile ISP's as she roams with her mobile device.

5. CONCLUSIONS AND FUTURE RESEARCH

We have presented a proposal for the use of a context-aware security policy agent to customize the security services for a mi-service to the consumer's preferences. In addition, this customization allows accounting for the mobile device's available computing power and the consumer's movement into a dangerous area with a higher number of attackers, where more powerful security mechanisms are needed. The use of a CASPA is a form of service personalization that studies have shown is attractive to consumers (Ho and Kwok, 2003). For future research, we would like to prototype the CASPA to study performance characteristics and refine our approach. Another area of interest is to develop a technique that would automatically and accurately determine the nature and extent of dangerous areas in a mobile network.

References

Chae, M. and Kim, J. (December 2003), What's So Different About the Mobile Internet?, *Communications of the ACM*, Vol. 46, No. 12.

Dinsmore, P. et al, 2000, Policy-Based Security Management for Large Dynamic Groups: An Overview of the DCCM Project, proceedings, *DARPA Information Survivability Conference and Exposition, 2000 (DISCEX'00)*, Vol. 1, pp. 64-73.

Duflos, S., 2002, An Architecture for Policy-Based Security Management for Distributed Multimedia Services, proceedings, *Multimedia '02*, Juan-les-Pins, France.

Ghosh, A.K. and Swaminatha, T.M. (February 2001), Software Security and Privacy Risks in Mobile E-Commerce, *Communications of the ACM*, Vol. 44, No. 2, pp. 51-57.

Ho, S.Y. and Kwok, S.H. (January 2003), The Attraction of Personalized Service for Users in Mobile Commerce: An Empirical Study, *ACM SIGecom Exchanges*, Vol. 3, No. 4, pp. 10-18.

International Organization for Standardization, IS0 7498-2, Information Processing Systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture; http://www.iso.org/

International Telecommunication Union Telecommunication Standardization Sector (ITU-T), Recommendation X.800, Security Architecture for OSI; http://www.itu.int/rec/recommendation.asp?type=items&lang=e&parent=T-REC-X.800-199103-I

Josang, A. and Sanderud, G., 2003, Security in Mobile Communications: Challenges and Opportunities. Australasian Information Security Workshop (AISW2003), *Conferences in Research and Practice in Information Technology*, Vol. 21, C. Johnson, P. Montague and C. Steketee, Eds.

Joshi, J. et al (February 2001), Security Models for Web-Based Applications, *Communications of the ACM*, Vol. 44, No. 2, pp. 38-44.

Mallat, N., Rossi, M., and Tuunainen, V.K. (May 2004), Mobile Banking Services, *Communications of the ACM*, Vol. 47, No. 5, pp. 42-46.

OASIS, eXtensible Access Control Markup Language; http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

Varadharajan, V., 1990, A Multilevel Security Policy Model for Networks, proceedings, *Ninth Annual Joint Conference of the IEEE Computer and Communication Societies (INFOCOM 90)*, Vol. 2, pp. 710-718.

Ventuneac, M., Coffey, T., Salomie, I., 2003, A Policy-Based Security Framework for Web-Enabled Applications, proceedings, *1st International Symposium on Information and Communication Technologies*, pp. 487-492, Dublin, Ireland.

Yee, G. and Korba, L., 2005, Negotiated Security Policies for E-Services and Web Services, proceedings of the *2005 IEEE International Conference on Web Services (ICWS 2005)*, San Diego, California.

[1] NRC Paper Number: NRC 48236

CoLoS -A SYSTEM FOR DEVICE UNAWARE AND POSITION DEPENDENT COMMUNICATION BASED ON THE SESSION INITIATION PROTOCOL

Odej Kao and Stefan Lietsch
University of Paderborn Paderborn
Center for Parallel Computing
Fuerstenallee 11 33102
Paderborn, Germany
{okao, slietsch} @uni-paderborn.de

Abstract In this paper we present a new system that allows users to communicate easily and comfortably. It supports users by integrating several services such as location awareness and device independence. The user is for example not bound to one device nor must he know the exact address of the desired communication partner. He is also supported by all the information the system has or can generate from all of its services. The Session Initiation Protocol is used and extended in order to provide a scalable, secure and efficient platform for the communication system and its services. The basic functionalities are shown in a prototypical implementation.

1. INTRODUCTION

Nowadays communication means more than just talking to each other or chatting. The focus lies on exchanging information where neither the distance nor the nature of the communication partner is an issue. There are many ways to communicate with people but also with machines e.g. computers and there are more to come with growing global inter connection. Two main problems arise from this development. On the one hand a user needs several devices to use the different ways of communication and on the other hand the exact address of the opponent needs to be known to establish a connection through a certain media. In addition the user must know or at least try out the media-related address on which his partner is reachable.

To solve these problems a system with two main functionalities is desirable: On the one hand, it should act as a translator between the different types of media and on the other hand it should provide something like an extended phone book in which all communication relevant and additional data is stored. These two components enable a user of the system to communicate with every potential partner without knowing his exact address nor being bound to one specific device. Furthermore, the system can merge all the data from different types of media it has and possibly gets additional input from e.g. positioning devices. So it can generate useful information to support the user. By combining these two aspects (uniform communication and user support) we created a system that extends the existing ones presented in chapter 2 to be a modular, scalable, secure and user centered communication platform.

This paper is structured as follows: After the Introduction and Motivation in section 1, we give an overview on existing communication systems and comment on location applications (section 2). In section 3 we outline the difference between the existing systems and our concept and present some possible features. Afterwards we describe the concept of the CoLoS and its components in section 4 and subsequently we go into the prototypical implementation of our concept (section 5). Finally we present a brief conclusion and point out which future work needs to be done.

2. RELATED WORK

Recently different approaches towards personalized, device independent and user supporting communication systems were proposed. There are two main research directions: One aims at maximising reachability independent of the used device. The other direction focuses on the personalization of the system. We aim to integrate both directions and add a new aspect namely the localization of the user. In the following we introduce three representative systems and show their advantages and problems and afterwards give a short comment on the localization aspect.

The Mobile People Architecture (MPA) described in [Maniatis et al., 1999] is the base of most of the device independent communication systems. It is the first development of an open concept for a device independent communication application ensuring best possible reachability of its users. By extending the traditional layer model by a Personal Layer the reachability is mainly achieved. This layer represents the person itself by managing all owners devices and their reachability. Thus communication requests no longer go directly to the devices but to the user (resp. to his representative, the so-called Personal Proxy). This proxy decides how to proceed with the incoming request, routes it to the corresponding device and manages the communication. This so-called Personal Level Routing is the main point of the MPA, but it brings one bottle-neck namely the Personal Proxy. All the communication has to go through the proxy although there could be a better or faster way. This problem is being solved in the Iceberg Architecture [Wang et al., 2000]. The system is based on the MPA but has one major difference: it no longer has decentralized proxies for every user but concentrates many proxies in centralized units called Iceberg Points of Presence (IPoPs). These IPoPs have interfaces to many access networks (e.g. telephony, cellular, internet) and are interconnected by fast network connections. This ensures that all communications can be routed on a fast and direct way. There is also a billing unit in the system to charge the users for certain services. In conclusion the Iceberg Architecture is a highly developed system to enable device independent communication but it does not support the user beyond this functionality and is limited to communication services.

The Integrated Personal Mobility Architecture (IPMoA) [Thai et al., 2003] takes a slightly different approach to ensure the device independency and mobility of the user. It does not primarily focus on the reachability but on the mobility of its users. The user is able to access all his data and applications from every remote location and with every available device. By including communication applications a device independent communication is possible. The whole system is based on agents that commute between the home and the foreign network and exchange the data between them. Thereby a high level of personalization can be reached but since nearly all data must be fetched from the home network it may have problems especially with time-sensitive and synchronous applications, respectively.

In the field of localization techniques there are many different approaches. One main research field is localization by determining the positions of all kind of mobile communication devices (e.g. GSM phones or WIFI devices) as proposed in [Youssef et al., 2003] and [Zimmermann, 2001]; another research direction is to use proprietary short range radio techniques based on bluetooth or infrared to locate the users of the system as exemplarily proposed in the Active Badge System [Want et al., 1992]. Since we want to support as many different localization systems as possible we don't want to commit ourselves to one technique or direction. We plan to integrate the positioning applications independent from the underlying mechanisms. This is why we just reference to some exemplary systems and focus on supplying an extendable communication and localization platform.

3. WHAT'S NEW?

All the systems presented above have different kinds of information about its users e.g. reachability, different addresses etc. This information is used to provide the functionalities of the communication systems. Our approach is to take all the data gathered by the different communication systems, add some additional data e.g. from location systems, and generate information that supports the user far more than possible with existing systems. Thereby we enable a platform for device independent communication and its personalization which combines the advantages of both directions presented in Section 2. Some possible services of this combined system are:

Find communication partners in your proximity. One self-evident service is to announce possible communication partners or friends which are detected near the location of the user. Possible scenarios for this service could be exhibitions where interesting exhibitors nearby are indicated to the user or the sign-posting to one specific person.

Discover the cheapest or fastest connection. Another possible service is to suggest the best and/or cheapest network connection for the users current position. Furthermore automatic connection handover mechanisms are possible.

Hints on services close to the user. The system can point nearby services out to the user. This could be communication services like a locally bounded NetMeeting conference or non-communication related services like a public printer or fax machine.

To make sure that the services above and other new ideas work properly, are accepted by the users and don't cause security problems some architectural requirements must be fulfilled: Extendable and easy to integrate, Hardware independent and portable, Transparency, Easy to use, Optionality and privacy, Security of the data.

We will introduce some exemplary functions and features of the proposed Communication and Location System (CoLoS). There are more to come since the platform is expandable and scalable. One function we already mentioned is the communication with users whose addresses are not exactly known. Our system finds the desired user by any known information (e.g. name, email, preferences) and gives choices if more than one match exists. Another feature is the communication with incompatible devices. That is that two people with different devices (e.g a mobile phone and an instant messenger) can communicate without recognizing the incompatibility. The third and very important feature of our platform is the configurable information search engine. It can access all the data gathered by the different systems and search for useful information to support the user by applying different rules on the database. In this way, all information available in the common database can be used to find new information that supports the user. This approach removes the barrier of incompatible storage used for the different communication services. We are now able to use all available resources to provide valuable information to the user.

4. CoLoS CONCEPT

Figure 1 shows the two sides of the developed system and their main components. The CoLoS client allows the service utilization and notifies the user about new incoming information. This is provided by a GUI fitted to the particular device. Already existing applications are integrated through the so-called Client Interface which can be seen as an universal interface. The data exchange between server and client is handled by the CoLoS Connection on the client and the Controller on the server side whereas the data is transmitted in packets of the CoLoS/SIP protocol. One of the main server components is the User Register where all the data is stored in a fast database. This data can be accessed by the Decision Engine to combine it following pre-specified rules in order to create useful information. The last main component is the Communication Dispatcher and its Communication and Translator Modules. It enables the system to translate between incompatible types of media transparently. In the following we describe the main components of the CoLoS server and client.

4.1 Server Modules

The **Controller** is the main server component. It decides by the type of an incoming message how to proceed with it. It passes the information of a message including a location update to the User Register in order to update the database. The Controller also decides by means of the data from the User Register by which device a user is reachable and if an incoming communication request can be fulfilled (with or without translation).

The **User Register** can be seen as an extended phone book. It stores all user data irrespective of its origin. Some exemple data are: addresses for each communication device, availability and preference of communication ways, current location, profile of the user, a buddy list, public key etc. The list of stored data can be extended arbitrarily to cover all useful information about the systems users.

The **Communication Dispatcher** is invoked by the Controller if a user requests a communication with another user who is not reachable under a compatible device. That is for example if user A wants to use an IP-Phone and user B only has his ICQ Messenger enabled. The controller recognizes this incompatibility and passes all messages concerning this communication to the Communication Dispatcher. It determines the corresponding Communication and Translation modules (in this case an IP-Phone and a messenger interface and a text-to-speech/speech-to-text translator), translates the messages and passes them back to the controller. Afterwards the messages are sent to the receiver under the address of the initial sender so that the whole translation action is transparent for the users.

The **Decision Engine** tries to find out which services can be offered to the users. This is a major functionality of the CoLoS since all available data is taken into account. It therefore applies rules on the User Register after every change in the database to look for new and useful information. One possible rule could be: "Check if a user in the users buddy list is in his proximity after he changed his position". If one or more matches are found the result is passed to the Controller. The Controller then generates messages containing this information and sends them to the correspondent users. While the database is growing and is changed more often with an increasing amount of users new mechanisms must be found to restrict the search to concerned fields of data. For example a strategy based on the current location of the user is thinkable. Additionally rules can be defined and added while the system is running.

4.2 Client Modules

The **CoLoS Connection** is the correspondent to the Controller on the server side. It gets commands and communication data from the GUI or the Client Interface, packs them into CoLoS Protocol packets and sends them to the Server. When a new packet is received the CoLoS Connection decides by its type what to do with it. For example an incoming message with the type "User Information" is passed to the GUI where it is displayed correspondingly.

The **Client Interface** is the bridge between existing communication and location applications and the CoLoS. The Interface notices which of the registered applications are started and announces this through the CoLoS Connection to the server. Furthermore it intercepts connection requests to pass them to the server for further processing. Incoming messages are handed over to the corresponding application and the whole communication process is monitored for faults and interruptions to enable a quick solution. Since the Client Interface acts transparently the user can continue to work with his applications as usual but also has the advantages of the CoLoS. The interface can also be utilized to get Location data from positioning applications. This functionality must be controllable by the user at all time to avoid an unwanted surveillance.

By the **GUI** a user can access all functionalities offered by the CoLoS easily and quickly. It for example alerts him on incoming requests or offers a "friend list" where he can save contacts he often uses. Also searching and security functions are integrated. All settings made by the GUI are sent to the server and stored in the user Registry.

4.3 CoLoS Protocol

The data exchanged between the CoLoS server and the clients is, as mentioned above, encapsulated in packets of the CoLoS Protocol. This ensures an efficient and secure data transmission. The CoLoS Protocol is implemented as an overlay protocol based on TCP/IP / SIP. It has several control fields (e.g. sender and receiver address, type of the message, additional options) and one optional data field.

4.4 Exemple CoLoS Interaction

For a better understanding of the system and the interaction of its components we describe one example process in the CoLoS. The scenario is that user A has the CoLoS client and a positioning device enabled and changes his location. An other user B, who is on As friends list, is nearby As new location. User A is notified and can chose between different options.

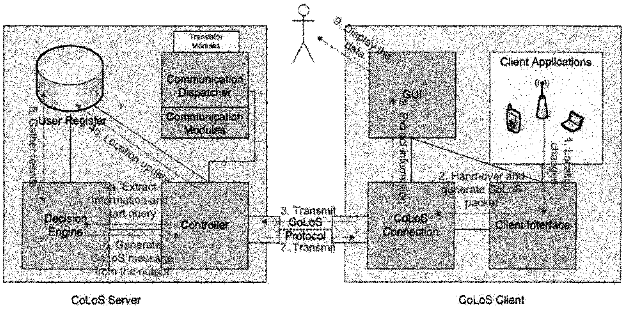


Figure 1. Processing of location updates

This process is depicted in Figure 1 and takes place as follows: The localization device realizes the change of positions and passes this information to the Client Interface (1.). The Client Interface gathers it and generates location update messages in fixed intervals (2.). This avoids an overload of the system due to too fast or too many location updates. The messages are passed to the CoLoS Connection where they are packed in CoLoS Protocol packets and transmitted to the Controller on the server side (3.). The controller receives the packet and extracts the type and the information included. Due to the type of the message the Controller decides to initiate a query through the Decision Engine (4a.). Simultaneously the position update is send to the User Register (4b.). The Decision Engine selects the rules corresponding to the received information and applies them to the User Register to search for helpful information (5.). In this case the rule "Find friends of User A that are closer than 50 m" could be selected and used to start a query. If the query has one or more results, this information is passed back to the Controller (6.). There a CoLoS Protocol packet is generated and transmitted to the client (7.). The ColoS Connection extracts the data and analyzes the type (8.). Afterwards the information is passed to the GUI where it is visualized and different choices of actions are presented to the user (9.).

This process is already implemented in the Prototype, as described in the following paragraphs, and it works well in a closed environment. It is ideal for the interaction in the CoLoS system.

5. CoLoS PROTOTYPE

To realize the components and functions of the presented concept we decided to base our system on the Session Initiation Protocol (SIP). This gave us the chance to use some of the well-tested and approved functions of the SIP and extend it to our needs. The message transmission mechanisms for example fully satisfy our requirements and already have security and recovery functions built in. Moreover, many communication applications already implement the SIP and can therefore easily be integrated into our platform. More information about the Session Initiation Protocol can be found in [Rosenberg et al., 2002].

As shown in Figure 2 we integrated the CoLoS components into the SIP architecture. Our system uses SIP to transport its messages, takes advantage of its routing algorithms and ensures the safety of the message transmission through its security mechanisms. The server side of the CoLoS is combined with the SIP-Proxy-Server and the Client uses the SIP-User-Agent to send and receive its messages. In both, the client and the server parts of the SIP, messages concerning the CoLoS are recognized and forwarded to the corresponding CoLoS component. This is where the content of the message is processed and if needed a response is generated and sent back using SIP mechanisms. Since the SIP-Proxy-Server is designed only to forward requests, we developed an additional component called SIP Client Simulator that simulates the server to be a client. This allows the CoLoS server to send unsolicited messages to the users device to announce, for example, a friend in his proximity.

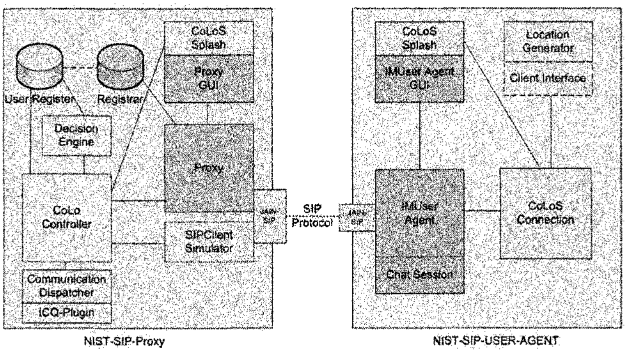


Figure 2. Realization of the CoLoS using the SIP

Our prototype is designed to show the basic features of the CoLoS. It already has an interface for ICQ/AIM Messaging, the SIP Messenger and a simple Location Generator that simulates a localization device connected through the Client Interface. On the server side we implemented the Communication Dispatcher with modules to translate SIP Messenger messages to ICQ messages and reverse, a Decision Engine that can generate information and pass it back to the Controller and a User Register utilizing a MySQL database which interacts with the registrar service of the SIP Proxy. To exchange data between client and server we developed a data format called CoLoS Protocol that is embedded in the SIP message. This encapsulates all CoLoS relevant data. The prototype was tested in different environments and gives a glimpse on what a complete system is able to do.

6. CONCLUSION AND FUTURE WORK

In this paper we proposed a design for an integrated and user supporting communication system. We analyzed the requirements and came forward with proposals for possible services. Regarding the current, fast development in communication technologies we designed a system that is not limited to existing applications. In fact we developed an open platform which is able to integrate existing and future technologies and to link them seamlessly. This is achieved with a modular structure and the use of standardized and application independent protocols. Thus the system can be extended by arbitrary applications to serve its users as a transparent, easy to use and secure communication base.

The prototype we presented was implemented to demonstrate the main features of the CoLoS. In a limited surrounding we can claim the CoLoS prototype works well and efficient. Since the SIP is already tested and approved in large-scale network environments we only have to test and scale the CoLoS specific features.

In the future the simplification and personalization of communication will gain more and more importance since the number of ways to communicate increases steadily and many users do not want or simply cannot take care of the maintenance of all the media. Additionally, negative aspects of the expanding communications world, such as unwanted spam, could be effectively fought by intelligent communication platforms as proposed in this paper.

References

Maniatis, P., Roussopoulos, M., Swierk, E., Lai, K., Appenzeller, G., Zhao, X., and Baker, M. (1999). The mobile people architecture. *Proceedings of the USENIX Symposium on Internet Technologies and Systems, October 1999.*

Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and Schooler, E. (2002). Rfc 3261 - sip: Session initiation protocol. Standard, IETF.

Thai, B., Wan, R., Seneviratne, A., and Rakotoarivelo, T. (2003). Integrated personal mobility architecture: A complete personal mobility solution. *ACM Mobile Networks and Applications (MONET) Special Issue: Personal Environment Mobility in Multi-Provider and Multi-Segment Network, Vol 8, Iusse 1, Feburary 2003.*

Wang, H., Raman, B., Biswas, R., Chuah, C., Gummadi, R., Hohlt, B., Hong, X., Kiciman, E., Mao, Z., Shih, J., Subramanian, L., Zhao, B., Joseph, A., and Katz, R. (2000). Iceberg: An internet-core network architecture for integrated communications. *IEEE Personal Communications (2000): Special Issue on IP-based Mobile Telecommunication Networks.*

Want, R., Hopper, A., Falcao, V., and Gibbons, J. (1992). The active badge location system. *IACM Transactions on Information Systems, Vol. 10, No. 1, January 1992, pp 91-102.*

Youssef, M., Agrawala, A., and U.Shankar (2003). Wlan location determination via clustering and probability distributions. *IEEE International Conference on Pervasive Computing and Communications (PerCom) 2003.*

Zimmermann, R. (2001). Lokalisierung mobiler geraete. *Seminar Mobile Computing ETH Zuerich 2001.*

A KEY-EXCHANGING SCHEME FOR DISTRIBUTED SENSOR NETWORKS

Hung Le Xuan, Sungyoung Lee, and Young-Koo Lee
Department of Computer Engineering,
KyungHee University, Korea
lxhung@oslab.khu.ac.kr, {sylee, yklee}@khu.ac.kr

Abstract: In order to achieve secure node-to-node communication in the distributed sensor networks, key management is the most important issue. However, due to limitations of sensor nodes in terms of energy, storage and communication bandwidth, this is a non-trivial job. Pre-distribution of secret keys is one of the most efficient ways. Currently, several key pre-distribution schemes for distributed sensor networks have been proposed. To our best knowledge, all of these efforts are on how to distribute shared keys in efficient manner. However, they do not consider the efficient path of node-to-node communication. In this paper, we propose a additional key-exchanging scheme for existing key pre-distribution scheme. We show that, by shifting secret keys to neighboring nodes, we reduce communication and computation overhead noticeably for the networks.

1. INTRODUCTION

Nowadays, distributed sensor networks (DSNs) are one of the most emerging technologies. Many applications such as distributed information gathering and distributed micro sensing in radiology, military, and manufacturing drive the research in sensor networks. However, DSNs differ from the traditional ad hoc network in several important areas. As consequently, security schemes for ad hoc networks can not be applied to DSNs such as public key scheme of Diffie-Hellman [3] or singly mission key.

One of the most important challenges of sensor network security is the design of protocols to bootstrap the establishment of a secure communication infrastructure, i.e. establishing a common key between two nodes so that they can communicate with each other in secure manner. The difficulty of the bootstrapping problem stems from the numerous limitation of sensor networks. In order to solve this problem, there have been three types of bootstrapping schemes: trusted server scheme, self-enforcing scheme, and key pre-distribution scheme [2]. The trusted-sever scheme depends on a trusted server for key agreement between two nodes, e.g. Kerberros [4]. This type of scheme is not suitable for sensor networks since there is usually no trusted node in sensor networks. The self-enforcing scheme depends on asymmetric cryptography. However, this scheme is unfeasible for sensor networks due to energy and memory limitation of sensor nodes. The other scheme, key pre-distribution seems to be the most suitable. In this scheme, key information is distributed among all sensor nodes prior to deployment.

Eschenauer and Gligor [1] recently proposed a random key pre-distribution scheme to address the bootstrapping problem. In this paper, we refer this scheme as the basic key pre-distribution scheme that we would like to improve performance. The operation of this scheme is briefly described in section 2. However, the problem of this scheme is that after path-key establishment phase, when two sensor nodes would like to transmit data through secure link, the message may travel along a long path before reach the destination. We name this as long-way exhaust problem of all key pre-distribution schemes. In order to solve this problem, we propose an additional key-exchanging phase to the basic scheme. The main idea is that after path-key establishment phase, each node broadcasts a notification message through the entire the network looking for the key with its neighbors. If such a node exists, it shifts the key to the broadcasting node along a secure path. After key-exchanging phase, every node shares at least one common key with each of its neighboring nodes. By shifting common key to two neighboring nodes, every node can find a shortest route to another node in the sensor networks.

The remaining paper is organized as follows. We first present an overview of the basic scheme in Section 2. Section 3 describes our key-exchanging scheme. We analysis our scheme and compare to the basic scheme in Section 4. We also give some discussion in Section 5. Section 6 concludes the paper and figures out some issues for our future work.

2. BASIC RANDOM KEY PRE-DISTRIBUTION SCHEME.

In [1], Eschenauer and Gligor proposed a Random Key Pre-distribution scheme based on probability model. This scheme is including three phase: key-predistribution, shared-key discovery, and path-key establishment.

Key Pre-distribution phase is processed before network deployment. A key pool S is created with keys. Each node randomly picks m keys from this pool and stores them in its memory. This set of m keys is called the node's key ring. The number of keys in the key pool, |S|, is chosen such that two random subsets of size m in S will share at least one key with some probability p.

After the sensor nodes are deployed, a key-setup phase is performed. During this phase, each node attempts to find out which node it shares a key with. To do this, every key is assigned with short identifier prior to deployment, and each node broadcasts this set of identifiers. If such a key exists, the key is used to secure the communication between these two nodes.

After key-setup is complete, a connect graph of secure link is established. Nodes can then setup path keys with their neighbors with whom they do not share keys. If the graph is connected, a path can always be found from a source node to any of its neighbors. The source node can then generate a path key and send it secure via the path the target node.

In this scheme, the authors attempt to provide high connectivity with less required memory, regardless to efficient communication later on. Assume that there are two neighboring nodes communicate with each other but they do not have any shared key. According to this scheme, in order to guarantee secure communication between these nodes, packets must be sent through path-keys which have been formed. It's obvious that such path is usually a long-way communication and consumes much energy of sensor nodes. How can we shorten this path while still guarantee secure communication between end-to-end nodes communication? In this paper, we solve this problem by using an additional phase, the key-exchanging phase.

3. A KEY-EXCHANGING SCHEME

In the basic scheme, any two neighboring nodes need to find a path-key in order to establish a secure link to transmit their packets. These paths are usually not efficient for routing protocol in terms of energy consumption and end-to-end delay. Thus, we propose a modification to the basic scheme where key-exchanging phase is additionally performed after path-key establishment. By shifting keys to neighboring nodes, we significantly increase the energy efficiency while still maintain original security of the basic scheme.

Figure 1 describes a simple case of long-way exhaust problem of the basic scheme. Assuming that after path-key establishment phase, network graph connection is presented as Figure 1. Considering that two nodes A and H would like to communicate to transmit packets via a secure communication. Thus, node A first has to send to node G which shares a common key . Node G then forwards to node E by encrypting the message with shared-key , so on and so forth until the message reaches the destination node H. As the results, the message travels along the path A-G-E-F-H. This long way costs much communicational and computational cost of sensor nodes for transmission, reception, key verification, message encryption, etc. Our approach solves this problem by establishing a secure link between G and H so that A can find the shortest path to H. In other words, our approach supports every node to find the shortest path to the destination.

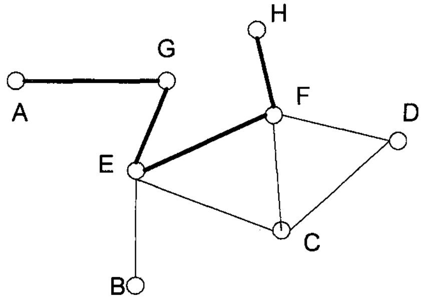


Figure 1. A Long-way exhaust problem of the basic scheme

3.1 Description of the Key-Exchanging Scheme

Notation	*Description*
n_i	Sensor node *i*
id_i	Identifier of node *i*
nonce	Random nonce value
K_{AB}	Private Key shared between A and B $(K_{AB} = K_{BA})$
$E(K, M)$	Encryption message *M* with key *K*
$\|\|$	Concatenation operation

Table 1. Notation used in Key-Exchanging Scheme

The operation of the key-exchanging scheme is similar to that of the basic scheme, different only in the additional phase as illustrated in Fig.2. In this operation flow, key-exchanging phase is performed after path-key establishment phase. Table 1 describes the notation used in our scheme.

3.2 Key-Exchanging Phase (Additional step)

The protocol for the *key-exchanging phase* is as follows:

$$n_i \rightarrow broadcast\ \ id_i \,||\, \{id\}_{i'}$$

Firstly, each node n_i includes its identifier id_i along with all identifiers of its neighboring nodes $\{id\}_{i'}$ in the "*hello*" it broadcasts after path-key establishment. In order to simplify the computational and communicational cost, message is transmitted without any encryption.

$$n_i \leftarrow n_j \quad id_{n_j} \,||\, E(K_{(\gamma-1)\gamma}, K_{ji'} \,||\, nonce)$$

We assume that node n_j posses a shared key $K_{ji'}$ with one of n_i's neighboring nodes, say $n_{i'}$. n_j then replies to n_i along a secure path $\Gamma = \{n_j, v_1, v_2, ..., v_l, n_i\}$ in sequent order[1]. Here, $K_{(\gamma-1)\gamma}$ is a secret key between two neighboring nodes $x_{\gamma-1}$ and x_γ on the secure path Γ, i.e. $(x_{\gamma-1}, x_{\gamma-1}) \in \{(n_j, v_1), (v_1, v_2), ..., (v_l, n_i)\}$. n_j then marks $K_{ji'}$ as a *exchanged-key*. This is important since after *key-exchanging phase*, every node should remove all *exchanged-keys* to release their memory.

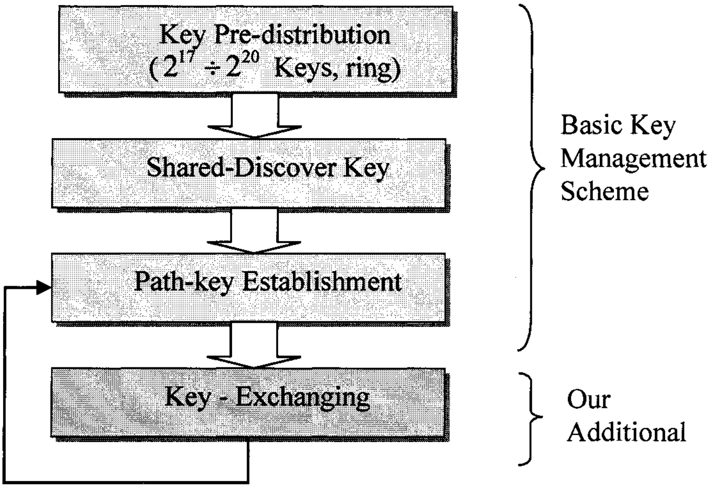


Fig. 1. Work Flow of Key-Exchanging Scheme

[1] To make key-exchanging more secure, n_j may disassembles $K_{ji'}$ into a set of $K_{ji'} = \delta_1 \oplus \delta_2 \oplus ... \oplus \delta_n$ and sends each number δ_i through different secure paths. When n_i receives all $\{\delta_1, \delta_2, ..., \delta_n\}$, it infers the key $K_{ji'}$ by assembling $K_{ji'} = \delta_1 \oplus \delta_2 \oplus ... \oplus \delta_n$.

After key-exchanging phase, path-key establishment phase is performed again. The purpose of this repeated step is to clear out all unnecessary path-key and setup a new path-key for the entire network.

4. ANALYSIS

In the basic scheme [1], Eschenaeur and Gligor used Random Graph theory to analyze DSN connectivity. A random graph *G(n,p)* is a graph of n nodes for which the probability that a link exists between two nodes is p. In a large sensor network with size *n*, *p* denotes the probability that two neighboring nodes share common key or key information, which we call local connectivity. Let P_c be the probability that the graph is connected, which we call global connectivity. Erdös and Rényi [11] provided a theory how to determine p so that P_c is almost 1 (i.e. the graph is almost surely connected).

Erdös and Rényi [11] showed that, for monotone properties, there exists a value of p such that the property modes from "nonexistent" to "certain true" in a very large random graph *G(n,p)*. The function defining p is called the thresh hold function of property. Given a desired probability P_c for graph connectivity, the threshold p is defined by:

$$P_c = \lim_{n \to \infty} P_r[G(n,p) \ \ is \ \ connected] = e^{-e^{-c}} \tag{1}$$

where $p = \ln(n)/n + c/n$ and c is any real constant (2)

Eschenaeur and Gligor [1] analyzed that given n, they can find p and the expected degree of node (i.e. the average number of edges connecting that node with its graph neighbors) *d = p(n-1)* for which the resulting graph is connected with desired probability P_c. We now prove that:

Theorem 1

After the *additional key-exchanging phase*, given *n* and the expected degree of node *d*, the probability P_c that the network graph is connected is always greater than that of the basic scheme (i.e. *key-exchanging* operation does not decrease the probability for graph connectivity).

Proof: Given *n* and *d*, we assume that P_c and P_c' are probabilities that the

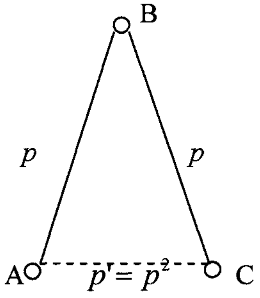


Fig. 2. The *Key-Exchanging* operation shifts secrete K_{AB} from B to C.

network graph is connected before and after *key-exchanging* operation, respectively. We now prove that $P_c' > P_c$.

Since p is the probability that a shared key exists between two nodes before *key-exchanging* phase, thus after *key-exchanging* phase this probability becomes $p' = p^2$. To prove this, we assume that there are three nodes A, B, and C (C is neighbor of A) as depicted in Fig.3. A and B share a common key with probability p. B and C share a common key with probability p. Thus, after the secret key K_{AB} is moved from B to C, the probability that a shared key exists between A and C is $p_{AC} = p' = p_{AB} \cdot p_{BC} = p^2$.

Since every node receives a secret key shared with each of its neighbors and vice versa, the probability that a shared key exists between two nodes in the network is:

$$p'' = 2p'(n-1) = 2p^2(n-1) \tag{3}$$

Because typically the number of sensor nodes $n > 1$ and $c > 0$, then from (2) and (3) we infer:

$$p = \frac{\ln(n)}{n} + \frac{c}{n} > \frac{1}{2n} \approx \frac{1}{2(n-1)}$$

$$\Leftrightarrow 2p(n-1) > 1$$

$$\Leftrightarrow 2p^2(n-1) > p$$

$$\Leftrightarrow p'' > p \tag{4}$$

Since P_c is directly proportional to p (or d), then from (4) we infer $P_c' > P_c$, i.e. *key-exchanging* operation does not decrease the probability of graph connectivity.

5. DISCUSSION

Obviously, shifting shared keys to each pair of neighboring nodes give a dramatic advantage for secure routing of DSNs. Every node can find the best path to its target through secure links. Since we do not reduce the number of key pre-distributed in entire sensor networks, but it may be increasing after key-exchanging phase, it is evident that the connectivity of the network is increasing. Consequently, other properties of the basic scheme are still maintained. One of the most advantages of this scheme is that it can be applied to whatever existing key pre-distribution schemes.

However, this scheme brings out many issues such that how to keep communication overhead of key-exchanging phase as minimum as possible. Another issue is how key-exchanging operation guarantees that the key is lost due to packet lost during transmission. We leave these issues for our future work.

6. CONCLUSION AND FUTURE WORK

We presented an improved scheme over Eschenaeur and Gligor scheme. This scheme gives an additional step, *key-exchanging phase*. By shifting the common keys to each pair of neighboring nodes, we can reduce significant computation and communication overhead of node-to-node communication while still guarantees original security of the basic scheme. This scheme, however, can be applied for all existing key pre-distribution schemes to improve the performance of secure routing for sensor networks.

In this paper, we have proposed a dramatic improvement over the basic scheme. In future work, we will investigate how much communication and computation overhead for *key-exchanging* operation. We also study how much our scheme supports to reduce energy consumption and computational cost for secure routing compared with the basic scheme.

References

[1] Laurent Eschenauer and Virgil D. Gligor. A key-management scheme for distributed sensor networks. In Proceedings of the 9th ACM Conference on Computer and Communication Security, pages 41–47, November 2002.

[2] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," IEEE INFOCOM, March 2004.

[3] W.Diffie and M.E.Hellman New direction in cryptograph. IEEE Transaction on Information Theory vol. 22 pp.644-654. November 1976

[4] B.C. Neumab sbd T. Tso. Kerberos: An authentication service for computer networks. IEEE communications Magazine. vol 40. no. 8. pp. 102-114. August 2002

[5] H.Chan, A. Perrig and D.Song. A random key predistribution schemes for sensor networks. in IEEE Symposium on Security and Privacy. Berkeley, California, May 11-14 2003 pp.197-213

[6] Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In IEEE Symposium on Security and Privacy, May 2003.

[7] Wenliang Du, Jing Deng, Yunghsiang S. Han, and Pramod K. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In ACM CCS 2003, pages 42–51, October 2003.

[8] Donggang Liu and Peng Ning. Establishing pairwise keys in distributed sensor networks. In ACM CCS 2003, pages 52–61, October 2003.

[9] Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. In Seventh Annual ACM International Conference on Mobile Computing and Networks (MobiCom 2001), July 2001.

[10] D. W. Carman, P. S. Kruus and B. J. Matt,"Constraints and Approaches for Distributed Sensor Network Security," dated September 1, 2000. NAI Labs Technical Report #00-010,

available at http://download.nai.com /products/media/nai/zip/nailabs-report-00-010-final.zip

[11] J. Spencer, The Strange Logic of Random Graphs, Algorithms and Combinatorics 22, Springer-Verlag 2000, ISBN 3-540-41654-4.

[12] F. Stajano, Security for Ubiquitous Computing, John Wiley and Sons, New York, Feb. 12, 2002, ISBN: 0-470-84493-0, 267 pp.

[13] Haowen Chan, A. Perrig, D. Song. Key Distribution Techniques for Sensor Networks. Springer-Verlag 2004, ISBN:1-4020-7883-8. pp. 277 – 303

A Look Into the 4G Crystal Ball

Simone Frattasi[1], Frank H.P. Fitzek[1], Ramjee Prasad[1]

Center for TeleInFrastruktur (CTIF), Aalborg University,
Niels Jernes Vej 12, 9220, Aalborg, Denmark
{*sf*|*ff*|*prasad*}@*kom.aau.dk*

Abstract. The time for reflections and visions about the *Fourth Generation of Wireless Mobile Communication Systems* (4G) is getting closer to the X hour, therefore the research community has to finally declare how 4G will really look like. Besides the no-boundaries visions, in this paper we have a look into the 4G crystal ball, attempting to define the forthcoming system fusing both the user and the technology perspectives.

1 Introduction

The major reason for the "failure" of the *Third Generation of Wireless Mobile Communication Systems* (3G) has been the tremendous worldwide downturn in the economy that lead to a starvation of new investments and hence to delays in technological development, service roll out, etc. In some countries with high penetration of mobile services, this was also combined with a reward policy for releasing the spectrum usage rights that did not favor the investments in the *International Mobile Telecommunications* 2000 (IMT-2000) infrastructure. However, besides these difficulties, the evolution from the *Second Generation* (2G) towards 3G has not brought any substantial new service for the customer, leaving the business model largely unchanged. The well known services plus some additional ones are provided, but they may not be enough to encourage the customers to change their equipment. The lack of innovative and appealing services was encountered too late by the *3G Partnership Project* (3GPP). In the latest standards, an attempt was made to incorporate some advanced services into the 3GPP architecture such as the *Multimedia Broadcast and Multicast Service Center* (MBMS) in combination with the *IP Multimedia System* (IMS). Nevertheless, these smaller improvements were made without the possibility to adjust the access technology properly. Ultimately, it has to be underlined that the limited success of the new technology has also depended upon the cultural and social settings in which the new system has been deployed. Indeed, 3G has been more accepted in Asian countries than in Europe: in Japan, the teenagers share videos and books on their mobile phones in public places, whereas in Europe they experience the same exchange indoor, in their own rooms with their own TV-sets or computers.

The difficulties and the technical limitations of 3G [1], as well as the emergence of new mobile broadband technologies on the market, have brought universities and industries to a throughout reflection on the *Fourth Generation* (4G).

The latter is expected to have a highly eclectic structure, where many devices, networks and protocols interact in a complex way to support the increasing user demands. Basically, many prophetic visions have appeared in literature [2], presenting the forthcoming generation as the ultimate boundary of the wireless mobile communication without any limit in its potential, but practically not giving any designing rule and thus any definition of it. Recently, a first attempt has been done within the framework of the *Joint Advanced Development Enabling 4G* (JADE) project[1], where a pragmatic methodology [3], centered on a user-centric approach, has lead to the definition of the key features and the technological step-up to be undertaken in 4G [4]. This methodology answers critically to the MAGIC view of NTT DoCoMo [5] as well as to the somehow cloudy one of the *European Community* (EC) of connecting 'everything with everything' [6]: the technological possibilities derived from the heterogeneity of terminals and networks are tremendous if – and only if – they meet the customers' needs and requirements. Indeed, since the ultimate goal of the technology is for communication requirement of human beings and the service is, to some extant, exhibitions of human requirement for communication, the trend of the service provision will definitely impose its influence on the underlying architecture design [7]. This methodology is hence strictly oriented towards a use of the technology for better life conditions and fair societies, where the inner logic of the technological evolution is coordinated with the societal one.

In line with [3] and [4], "4G will be a convergence platform providing clear advantages in terms of coverage, bandwidth, and power consumption. Moreover, it will offer a variety of new heterogeneous services, from pop-up advertisements to location-based and interactive or on-demand ones – so called IP datacasting. All these characteristics will be supported by multi-mode / reconfigurable devices and the implementation of interworking ones". In order to achieve the previous outlined goals, in this paper we motivate the cellular controlled short-range communication architecture adopted in JADE and consequently extend the framework developed in [3] and [4] – the "User-Centric" System – introducing the issue of *cooperation*. In particular, we show an innovative cooperative geolocation scheme supported by such an architecture that combining long- and short-range location information enhance the location estimation accuracy with respect to the actual stand-alone cellular solutions. This scheme could hence support in the future *Cooperative Location Based Services* (CLBSs). Finally, an insight on the social dimension related to cooperative services in general is also discussed at the end of the paper.

The rest of the article is structured as follows: Section 2 describes the JADE system architecture; Section 3 presents the "Cooperative-User-Centric" System as the extension of the "User-Centric" System; Section 4 illustrates the proposed *Cooperative Localization Scheme* (CLS); and Section 5 discusses the social dimension related to the proposed framework. Finally, the concluding remarks are given in Section 6.

[1] The JADE project is a cooperation between SAMSUNG Korea and the *Center for TeleInFrastruktur* (CTIF), Aalborg University.

2 JADE System Architecture

The overall target is a cellular system that also supports short-range communications among the terminals. The rationale for introducing short-range communications is mainly due to two arguments: 1) The need to support *peer-to-peer* (P2P) high-speed wireless links between the terminals; 2) The need to enhance the communication between a terminal and the *Base Station* (BS) by fostering cooperative communication protocols among spatially proximate devices. The communication enhancement primarily refers to a higher link reliability, a larger coverage, a higher spectral efficiency and a lower power consumption thanks to the use of exclusive cooperative stations (e.g., *Relay Stations* (RSs) deployed by operators) or short-range communications among different mobile terminals.

The cellular system will be synchronous and a tight control will be imposed by the BS over the short-range communications among the devices associated with it. The wireless terminals will use the same air interface for long- and short-range communications. The expression "same air interface" means that the basic set of access technologies should be the same for links with the BS as well as P2P links. However, the access technologies should be tunable, such that a terminal can seamlessly adapt the transmission format to the cellular links and the short-range P2P ones, respectively. As an illustration, the same air interface implies that some transmission formats applied when a terminal transmits to the BS can be received also by another terminal. Nevertheless, the BS will apply more complex receiver algorithms and will thus be able to extract more data from the same transmitted message.

2.1 Cooperative Communication

Recently, much research effort has been put to understand and utilize the benefit of cooperative behavior in wireless networks [9]. The concept of *cooperation* introduces a new form of diversity where the terminals are less susceptible to the channel variations and shadowing effects. This results in an increased reliability of the communication and the extension of the coverage. Furthermore, whereas in voice networks the resources are dedicated for each user separately, in cellular controlled short-range data networks it is possible to group the users in clusters with the following advantages: a) Only the *Cluster Head* (CH) needs to have a dedicated channel to the BS, while the other terminals can communicate by using *unlicensed bands*, thus more bandwidth is not required; b) Due to the short range of the transmissions performed by the terminals to the CH, it is possible to reduce their power consumption and hence prolong their battery life.

In order to achieve the previous outlined goals, the work is mostly focused on the design of different cooperative mechanisms in the following layers of the protocol stack:

– *Physical* (PHY) layer. The research in PHY-layer cooperation is in a mere infancy, but it has already promised a great potential. Cooperative diversity protocols exploiting the feature of wireless broadcast medium have shown potential to achieve similar effect to conventional *Multiple Input Multiple Output* (MIMO) transmissions. However, theoretical analysis of capacity increase brought by cooperative behavior has been the main topic in the literature [10], while there have been few works on practical protocol designs to achieve the high diversity gain. This research area focuses on the design of coding, modulation, receiver algorithms and forwarding mechanisms such as *Amplify and Forward* (AF) and *Decode and Forward* (DF) in order to achieve high diversity gain in practical cooperative scenarios.

- *Medium Access Control* (MAC) layer. An important issue to be addressed is the formation of the cooperating group for different targets (coverage enhancement, energy consumption reduction, etc.), taking account of the inter-terminal channel conditions and the spatial distribution of the terminals. The design of protocols such as radio resource management and handoff has to be also addressed to coordinate transmissions between short-range links (links within cooperative groups) and long-range link (links between BS and terminals) so that the most appropriate links can be used by terminals with the least interference conditions.

3 A "Cooperative-User-Centric" System

The discussion about the system architecture adopted in JADE closes the circle of the methodological approach described in [3], fusing the user with the technological perspective, and addressing the technical step-up illustrated in [4]. In particular, the introduction of *cooperation* affects the "User-Centric" System, which in this section is extended to the "Cooperative-User-Centric" System.

Although we are used to think that the stars come as individuals because that's how our own Sun appears, this is not the norm. The evidence, instead, is that more than 85% of them are parts of multiple star systems, where the stars member revolve around a common center of mass under the influence of their mutual gravitational force. The most common multiple star system includes two stars and it is called binary star system. In some binary star systems, called close binaries, the stars are so close together that they can transfer matter to each other and change the way they look and evolve.

People, like stars, are seldom found in isolation. Therefore, along with the new architecture proposed in Section 2, we can represent 4G as a "Cooperative-User-Centric" System, where each user, according to [3] and [4], comprises its own planets (see Figure 1). Moreover, as multiple star systems when the stars are getting closer to each other raising up their velocity accordingly to their increasing mutual gravitational force (violet curve in Figure 1), the *user cooperativity* and consequently the services' performance obtained by clustering the users can be considerably enhanced (e.g., as shown in [8], the users' cooperation can lead to better services at lower prices). In the next section, a practical example of performance enhancement is described.

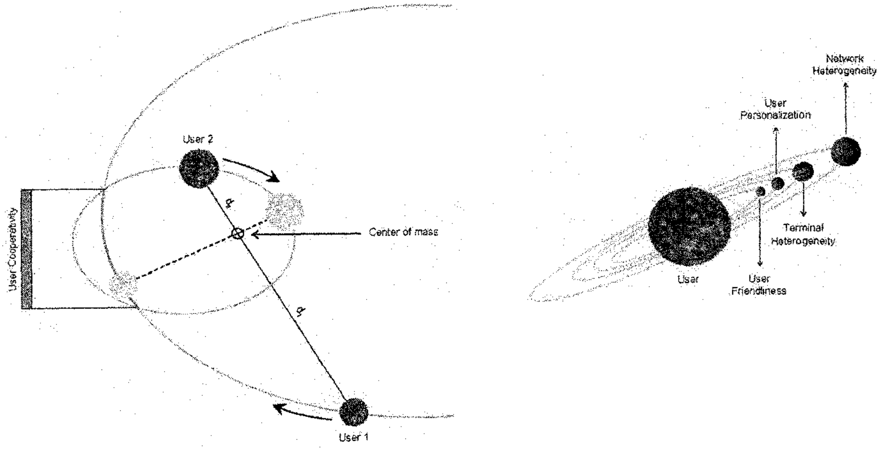


Fig. 1. The "Cooperative-User-Centric" System.

4 Cooperative Localization

Geolocation, or location estimation in terms of geographic coordinates of a *Mobile Station* (MS) with respect to a reference point in a wireless system, has gained considerable attention over the past decade, especially since the *Federal Communication Commission* (FCC) passed a mandate requiring cellular providers to generate location estimates for *Enhanced-911* (E-911) services with an accuracy of 100 meters for 67% of the cases [11]. This has boosted the research in the field of wireless location as an important public safety feature, which can also add many other potential applications to the future cellular systems [12].

In this section, we propose an innovative geolocation scheme that combines long- and short-range location information, respectively retrieved by mean of a *Hybrid Time Of Arrival/Angle Of Arrival* (HTA) technique in cellular networks and TOA technique in short-range networks, in order to enhance the location estimation accuracy with respect to the actual stand-alone cellular solutions. To the best of our knowledge, most of the existing studies in the literature tend to treat these positioning techniques separately for cellular and short-range networks. In the CLS, we practically suppose that the cellular system has supervised the formation of a short-range cluster and that the CH has relayed back all the measures of TOA corresponding to the measurements of relative distance obtained by each MS from the other members of the cooperative group[2] (see Figure 2). The TOA measures are then exploited in the location algorithm in order to weight the HTA measures associated to the users belonging to the cluster.

[2] The CH sends the data to the nearest BS thanks to a possible power control procedure exploited during the clusters' initialization phase.

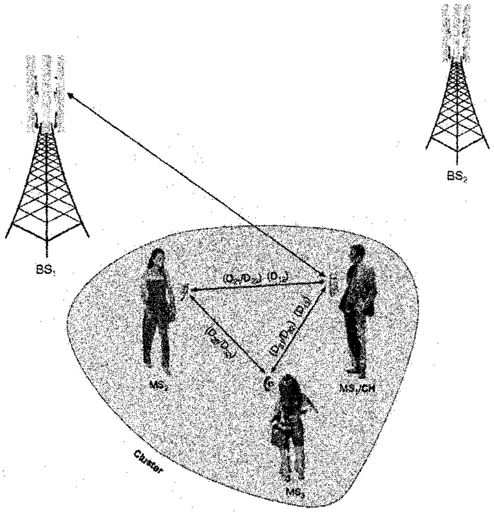


Fig. 2. Reference scenario.

System level simulations have been carried out developing a simulator in MATLAB. We have taken into consideration an urban-like scenario (micro cells of diameters around 2000 m), where a group of $1 \leq M \leq 8$ MSs is simultaneously in the range of $3 \leq N \leq 7$ BSs, where the CH is placed in the center of the cluster at a minimum distance of 100 m to the serving BS, being BS_1, and the other MSs are uniformly generated around it at a maximum distance of 100 m. The BSs are positioned in a two-dimensional plane with coordinates BS_1(0,0), BS_2(1732,1000), BS_3(1732,-1000), BS_4(0,-2000), BS_5(-1732,-1000), BS_6(-1732,1000) and BS_7(0,2000). We suppose that the speed of the users is very low and thus that, during the number of iterations considered for the location estimation, there is no relative movement between the cluster and the BSs, and within the cluster itself. Finally. we assume that the measurements are made on signals propagating via *line-of-sight* (LOS) paths; therefore, the estimation errors are small and primarily due to equipment measurement errors. The latter have traditionally been assumed to be normally distributed with zero mean and a small variance, respectively $c\sigma_t = 30$ m (the variances of the TOA measurement errors associated with different BSs are set as identical) and $\sigma_\alpha = 1$ deg. In particular, since each MS is carrying out a calculation on each short-range link, the measured distance between two MSs becomes more reliable and thus we set $c\sigma_{t_{rel}} = 3$ m. The values of the parameters used in the simulations are summarized in Table 1.

The performances of the proposed scheme are presented in the form of *Cumulative Distribution Functions* (CDFs) of the average location error, i.e., the estimation error resultant from the mean of the individual estimation errors, which has been computed based on 1000 independent runs.

Table 1. Simulation parameters.

PARAMETERS	VALUES
Number of BSs	1-7
Number of MSs	1-8
Distance BS/CH	100-900 m
Distance MS/CH	1-100 m
TOA error (long-range)	N(0,$c\sigma_t$=30)
TOA error (short-range)	N(0,$c\sigma_{t_{rel}}$=3)
AOA error	N(0,σ_α=1 deg)

Figure 3 shows the improvements introduced by the proposed scheme with respect to a stand-alone HTA due to the users' cooperation and the estimation of their relative distances. Considering three BSs, for example, and estimating the location of the users only with the HTA technique, the distribution of the average location error has a mean μ = 32.68 m, whereas, in case of eight cooperative users, μ drops to 22.69 m. This is due to the fact that the retrieved location estimates represent a certain configuration in the space, which has to respect the geometrical constrains of a polygonal structure (e.g., octagonal in case of eight users) based on the knowledge of the relative distances. As a consequence, the more cooperative users join the cluster the more binds the selected solution has to respect and the higher the accuracy is.

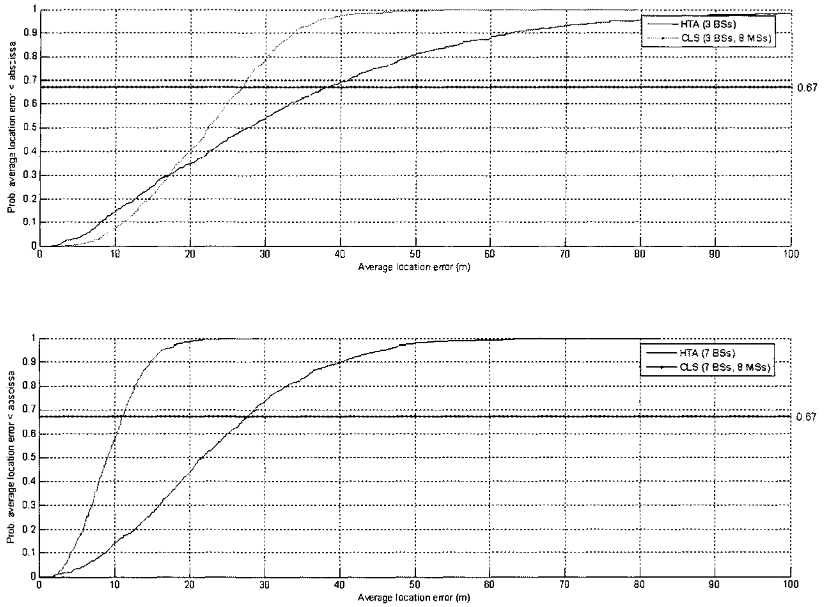


Fig. 3. Performance comparison between HTA and CLS.

The simulation results show that the CLS increases the location estimation accuracy with respect to a stand-alone HTA solution. Furthermore, when $N \geq 3$ BSs are available for location purpose the average location error achieves values comparable to the *Global Positioning System* (GPS). The proposed scheme hence demonstrates that the synergy between cellular and short-range communication systems can represent a valid architecture for 4G.

5 The Social Dimension of Cooperative Services

This section gives an insight on the potential social impact of cooperative services. The latter are mainly based upon three fundamental phenomena of social order:

1. *Group building.* Social groups on different levels (such as teenage groups, organized groups of football players, loosely organized groups of neighbors, which are relative stable over the time, or loosely organized and only for a short time stable internet groups) are the intermediaries in societies. Groups differ concerning what they have in common: spatial proximity (neighbors), interests (internet groups, business groups), age (teenage groups) and so forth. They are more or less durable over the time and more or less organized. Although the service provision is performed at the technical level, on the social level new groups might be established to lower the costs and offer a better quality of the services.
2. *Network.* As Manuel Castells outlined [13], fuzzy, spontaneous networks are increasing in importance. Cooperative services make intelligent use of these decentralized network of terminals, where the necessary but limited hierarchy is evolving out of the given situation and the decision concerning the master-terminal is drawn on the background of technical means and conditions seamless to the users. These new type of networks may be hence coined *situational hierarchy* and indicate the increasing relevance of new types of time-limited and functional networks. As a consequence, the success of cooperative services will depend not only on the potential resulting performance but also on the trust the customers have in the capability of the network to define a master-terminal that will be really able to coordinate.
3. *Cooperation/sharing.* Cooperation in the society is usually defined as a coordinated effort to reach mutual goals. The different reasons at the base of this coordination are: traditions, habits, emotions (like compassion), instrumental rational considerations (like efficiency, utility) or more normative rational ones (what one ought to do according a given social or moral norm). It depends very much on the context and the behavioral setting as well as on the motivation in the background and thus in the stability of the cooperation. Although the common sense makes us believe that the best motivation for cooperation is a personal good outcome or result, the concept of the self-interested actor that tries to maximize his own interests and profits is not always valid. In the cooperative use of the mobile phone, for example,

the rational interest of lowering costs and receiving a better quality of the services lies clearly in the realm of maximizing the profit. Nevertheless, also to become member of a group and to share not only material but symbolic resources (e.g., joint values, prestige, friendship, etc.), could be a strong motivation for utilizing such cooperative services. Indeed, we can imagine groups that would like to make clever use of them in order to watch videos or share other files that are usually more expensive. Therefore, ad-hoc communities based on agreements about what to watch together are then possible to set up. These services hence increase the cooperative behavior and empowers the consumer to make clever use of them. In general, the possibility for customers to deal in a creative manner with the technical possibilities is of great importance for the services' acceptance. In a way, the user terminal is not any more a bare medium to transfer information, but a social medium that helps to build groups and friendships.

6 Conclusions

Even though in the research community there is still uncertainty regarding the final silhouette of 4G, the objectives are quite clear and indeed the motto of the forthcoming system(s) is: "4G has to be cheaper and better than 3G". In this paper, as a result of a joint user-technology analysis, we have defined 4G according to the JADE project. The new system concept – the "Cooperative-User-Centric" System – has shown to potentially achieve the goals underlined in the 4G motto.

7 Acknowledgements

This work has been supported by Samsung Electronics, Co., LTD, Korea.

References

1. L. Zhen, Z. Wenan, S. Junde, H. Chunping, "Consideration and Research Issues for the Future Generation of Mobile Communication", in Proceedings of the 15th Canadian Conference on Electrical and Computer Engineering (CCECE), IEEE, vol. 3, pp. 1276-1281, Winnipeg, Manitoba, Canada, May 12-15, 2002.
2. E. Bohlin, S. Lindmark, J. Bjrkdahl, A. Weber, B. Wingert, P. Ballon, "The Future of Mobile Communications in the EU: Assessing the Potential of 4G", ESTO Publications, February, 2004.
3. S. Frattasi, H. Fathi, F.H.P. Fitzek, M. Katz, R. Prasad, "A Pragmatic Methodology to Design 4G: From the User to the Technology", in Proceedings of the 5th International Conference on Networking (ICN), IEEE, Reunion Island, France, April 17-21, 2005.
4. S. Frattasi, H. Fathi, F.H.P. Fitzek, K. Chung, R. Prasad, "4G: The User-Centric System", Mobile e-Conference (Me), Electronic Conference, August, 2004.

5. K. Murota, NTT DoCoMo, "Mobile Communications Trends in Japan and DoCoMo's Activities Towards 21st Century", in Proceedings of the 4th ACTS Mobile Communications Summit, Sorrento, Italy, June 8-11, 1999.
6. J. M. Pereira, "Fourth Generation: Now, it is Personal", in Proceedings of the 11 International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), IEEE, London, UK, September 18-21, 2000.
7. J. Yang, Z. Ping, "MUSE: A Vision for 4G Service and Architecture", in Proceedings of the 60th Vehicular Technology Conference (VTC), IEEE, Los Angeles (CA), USA, September 26-29, 2004.
8. S. Frattasi, B. Can, F.H.P. Fitzek, R. Prasad, "Cooperative Services for 4G", in Proceedings of the 14th IST Mobile & Wireless Communications Summit, Dresden, Germany, June 19-23, 2005.
9. A. Nosratinia, T. E. Hunter, A. Hedayat, "Cooperative Communication in Wireless Networks", Communications Magazine, IEEE, vol. 42, no. 10, pp. 74-80, October, 2004.
10. G. Li, H. Liu, "On the Capacity of Broadband Relay Networks", in Proceedings of the 38th Annual Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, November 7-10, 2004.
11. FCC, "FCC Acts to Promote Competition and Public Safety in Enhanced Wireless 911 Services", Washington, DC: WT Rep. 99-27, Sept. 15, 1999.
12. A.H. Sayed, N.R. Yousef, "Wireless Location", Wiley Encyclopedia of Telecommunications, J. Proakis, editor, John Wiley & Sons, NY, 2003.
13. M. Castells, "The Information Age: Economy, Society and Culture", vol. 1-3, Blackwell, Oxford, 1998.

DISCOVERING ARCHITECTURE FORMALISM OF GEO-LOCATED WEB SERVICES FOR NEXT GENERATION OF MOBILE NETWORKS

André Claude Bayomock Linwa [1] and Samuel Pierre [2]
[1,2] LARIM, École Polytechnique, C.P. 6079, succ. Centre-ville
Montréal (Québec), H3C 3A7, Canada

Abstract: Geo-located web services are web services offered in a particular geographical region. In mobile application design, a geo-located web service can be mapped to a set of mobile network location areas. As a mobile client roams in a mobile network, if he has a geo-located web service in execution progress at a supplier application server (SAS), he will lost its session in case when its current location is not covered by this SAS. With the next generation (third and up) of mobile networks, the geographical position of a mobile client will be sent back by a LoCation Server (LCS) to an application which requests it. As many geo-located web services will be deployed in the future, the great challenge for a mobile client will be to discover and maintain a geo-located web service when he is roaming. We propose a new system named Geo-Located Web Service Architecture (GLWSA) that aims to discover and maintain a geo-located web service with or without QoS at the nearest SAS of a mobile client current location. The GLWSA is a set of discover servers named GLWSMs (Geo-Located Web Service Manager) which are distributed in the topology. The GLWSA extends the UDDI and MLP protocols to add the GLWSM topology management and the thematic location of a mobile clients group, respectively. A thematic location consists of sending to a LCS, a chain of characters that represents a theme or a subject linking a group of mobile clients. In this paper, we present the GLWSA concepts and its mathematical formalism. Tests executed to evaluate the system performance prove that the GLWSA concepts are adequate to discover geo-located web services.

Key words: Geo-located services; Next generation mobile networks; Service discovery; Web services; Quality of services.

1. INTRODUCTION

In the development perspective of future mobile service applications, geo-located web services will be more and more deployed [5]. By definition, a geo-located web service is a web service offered in a specific geographical region. The geographical area restriction of a geo-located web service induces one main problem: how to maintain the service in execution progress when a mobile client leaves the geographical region covered by a geo-located web service? By adding to this problem the scalability deployment of geo-located web services, the great challenges for a mobile client that roams over a mobile network will be to discover and maintain a service in execution progress at the nearest supplier application server (SAS) of a mobile client current location. The service discovering and the service execution maintainability could be requested with or without QoS requirements.

An example of geo-located can be to do a virtual visit of a particular museum when the mobile client is in a specific geographical area.

In related work, the proposed architectures are either adapted to discover services where fixed clients are involved [6] and those which consider the mobility put the emphasis on code and agent mobility [7, 10] rather than data or task mobility to the nearest SAS (as the web services are service oriented and use a client/server model) based on the location context of the mobile client and maintainability of service execution.

To resolve these challenges, we proposed a new system named Geo-Located Web Services Architecture (GLWSA) that aims to lookup, publish a geo-located web service and coordinate the geo-located web service migration at the nearest SAS of a mobile client current location. The GLWSA also provides thematic location methods to locate a group of mobiles related by a theme or subject instead of locate a group mobiles by sending with a location request a list of mobile identifiers. The GLWSA is composed of a set of distributed Geo-Located Web Services Manager (GLWSM).

An example of thematic location is to determine the position of all *railroad trains of Via Rail Canada* which are in Montreal area. In this request, the "*railroad trains of Via Rail Canada*" is the subject or theme.

To discover a service with QoS (cost of service, network bandwidth and SAS utilization rate) and maintain a service execution with QoS, we defined a new mechanism that collects the network bandwidth of a particular geo-located web service and the SAS utilization rate at a specific GLWSM. For a specific service, the network bandwidth and SAS utilization rate are collected periodically in a particular GLWSM domain by sending a collect traffic message to all SAS that offer the concerned service in this domain. Each concerned SAS collects and sends back to the requestor GLWSM, the bandwidth and the SAS processor rate. Collected QoS parameters are used in the SAS selection and migration process.

The main contribution of this paper is to present the GLWSA concepts and its mathematical formalism. The mathematical formalism shows how the GLWSA properties, relations and functionalities (lookup, publish a geo-located web service and coordinate a geo-located web service migration at the nearest SAS) can be transform to algebra logics. Thematic location and the proposed mechanism to collect QoS are not in the scope of this paper.

The organization of this paper consists to present related work in discovering services, to explain the GLWSA concepts, to transform the system concepts in mathematical formalism, then evaluate the GLWSA system and give a brief conclusion.

2. RELATED WORK

Related work in the service discovery can be classified in two categories: classic and non classic protocols. Classic service discovery protocols are protocols which are commercially known and generally used in the service discovery process. SLP (Service Location Protocol) and Jini are the most popular classic service discovery protocols in literature. SLP (Service Location Protocol), a protocol developed by IETF, uses three agents: a UA (User Agent), an SA (Service Agent), and a DA (Directory Agent) [6]. On the other hand, Jini is a technology developed by Sun Microsystems for discovering services. Just as SLP, it involves three actors: the client, the service broker server, and the service supplier server [3]. However, SLP and Jini protocols are not designed for mobile clients and do not allow to discover a service based on a client's location context. Non classic discovery protocols are protocols proposed by research group to discover services but are not huge used. In [12], authors defined an architecture to locate mobiles and query databases based on their location context. The proposed architecture is a central middleware where services are published and discovered through a user service agent. The main component of the system is a location dependent service manager. The location dependent service manager controls the system. It analyzes the query and binds the pseudo-codes sent in the request to the correct predicates (e.g., 'nearest' can take the value 'five miles'). It verifies the granularity of the query, dispatches the request to the corresponding databases and returns the results to the client in the desired format.

In non classic discovering services protocols, we selected some relevant protocols.. In [7], an architecture named Application Module Request Broker (AMRB) is presented. This architecture enables clients to discover an application module (AM). The system is a distributed AMRB. Each AMRB has two main components: location and migration. Location and migration components allow to determine the current location of an application code and to migrate an application code to another host. The system proposes two kinds of migrations: host and code migration. The host migration uses a network location detection by sending periodic polling; code migration detects the migration of an AM and diffused the location change in a multicast process to all AMRB of the system.

In [11], an architecture for reconfiguration control and service provisioning platforms were proposed. This is a middleware system which mediates between a provider service called Value Added Service Provider (VASP) and the network resources in order to deliver services to end users according to their location context. The system informs the mobile clients of the communication cost as they change location area. Services are published and discovered through a service manager. Although this strategy seems similar to our approach, it fails to maintain a nearest service execution compared to the location context of the mobile client.

Other projects, such as Globe, use an architecture called "Globe Housing Service" [2]. This architecture allows locating the mobile users and services. Globe defines and implements distributed objects in a hierarchical topology tree. When a client looks up an object in a leaf node of a given location area, the object can be present or not. If it is absent, the system returns the contact address of the Globe object requested. This contact address redirects the request to the next node of the path tree. The procedure is repeated until the request is fulfilled. Globe is not adapted to discover services based on the location context of a mobile client.

3. THE PROPOSED SYSTEM CONCEPTS

3.1 UDDI protocol

The Universal Description, Discovery, and Integration (UDDI) protocol provides a standardized method for publishing and discovering information about web services [13]. Building a UDDI protocol is an industry initiative in order to create a platform-independent, open framework for describing services, discovering businesses, and integrating business services. The UDDI main objective is a web service discovery process in a service-oriented architecture. Basically, a UDDI protocol is associated with a single registry that can be distributed among many nodes. Each UDDI registry stores business information of a specific supplier or organization entity.

Conceptually, a business organization can register three types of information into a UDDI registry: white pages, yellow pages and green pages. The white pages are basic contact information and identifiers about a company, including business name, address, contact information, and unique identifiers. This information allows others companies or clients to discover a business organization web service based on the business identification. The yellow pages are information that describe a web service using different categorizations (taxonomies). These information allow others to discover a web service based upon its categorization (such as food business). The green pages are technical information that describe the behaviors and supported functions of a web service hosted by your business. These information include pointers to the grouping information of web services and where the web services are located [13].

The UDDI data structure is composed four elements: *businessEntity, businessService, bindingTemplate* and *tModel.* A *businessEntity* structure represents a business organization basic information. These information include contact information, categorization, identifiers, descriptions, and relationships to other businesses. A *businessEntity* contains one or more *businessService* structures. A *businessService* represents a description of one web service. This web service can contains one or many published methods. A *businessService* contains one or more *bindingTemplate* structures. A *bindingTemplat*e contains pointers to technical descriptions and the access point URL, but does not contain the details of the web service's specifications. A *bindingTemplate* contains an optional text description of the web service, the URL of its access point, and a reference to one or more *tModel* structures. À *tModel* is an abstract description of a particular specification or behavior to which the web service adheres. A *tModel* is linked to a WSDL (Web Service Description Language) document that determines specifically how to interact with a particular web service. Clients or others organizations can use the information pointed a *tModel*'s WSDL document to determine whether a web service is compatible with their requirements.

3.2 MLP protocol

The MLP protocol is application level protocol that aims to allow the interoperability of location requests with location servers LCS [1]. The MLP format language is developed using XML language. The MLP architecture has three main layers: service, element and transport [9]. On the lowest level, the MLP transport layer defines how XML content is transported. Possible

MLP transport protocols include HTTP, SOAP and others. The Element layer (second layer) defines all common elements used by the services in the service layer. The Service layer (top layer) defines the services offered by the MLP. The services are classified into five categories: standard location immediate service, emergency location immediate service, standard location reporting service, emergency location reporting service and triggered location reporting service. The standard location immediate service allows applications to request a single location response from the Location Server LCS. The request can also be served by asynchronously sending the location to the application until a timeout limit is reached. The emergency location immediate service is used when a mobile client initiates an emergency call. This service is mostly used by *911* applications. In the standard location reporting service, the position is periodically sent to an application until a timeout is reached. The emergency location report is used when the mobile network automatically initiates the position determination for an emergency call. The position and related data are then sent back to the emergency application. The difference between the emergency location immediate service and emergency location report is that the first one is initiated by the subscriber, while the second service is automatically triggered by the provider. The triggered location reporting service is an event-based service where the mobile subscriber's location is reported on the occurrence of a specific event.

3.3 Description of the proposed system

The GLWSA system is a distributed system composed mainly of three entities: the GLWSM server, the UDDIM database and the ClientInfosDB database (Figure 1). The GLWSM (Geo-Located Web Services Manager) server is the main component of the system. It keeps the implementation of the system functionalities and manages the geo-located web service operations (lookup, publication, coordination of service migration, service execution, mobile location, etc.). The UDDIM (UDDI for Mobiles) is an extension of the UDDI to adapt the UDDI registry to the mobility context by adding APIs and structures of the GLWSA topology (GLWSM nodes), the service agreement of a geo-located web service and the geo-located web service agreement per node. The ClientInfosDB database stores user personal data such as identification (name, address, username, password, etc.), equipment (mobile station identifier, phone number), subscription and quality of service data. A GLWSM interacts with three external entities: mobile client, the SAS (Supplier Application Server) and the LCS server. A mobile client is a person who has a mobile equipment (cell phone, PDA, etc.) and has subscribed to a specific geo-located web service. The SAS is the end server where a geo-located web service is really implemented and executed. The LCS is the location server which determined the geographical position of a particular mobile client. To interact with the LCS, the GLWSM uses a MLPe protocol (MLP Extension) which is an extension of the MLP protocol to realize the thematic location of a group of mobile clients.

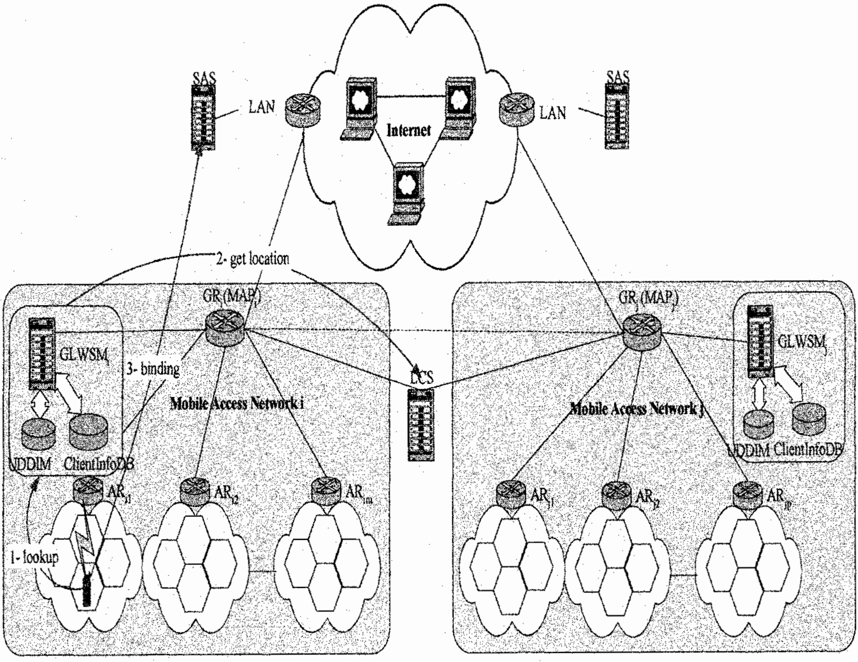


Figure 1. GLWSA architecture

3.4 GLWSA Topology

The GLWSA topology is bi-level hierarchical system. We chose a bi-level system to reduce the message exchange between GLWSMs in the service publication process and the propagation of the collected QoS data update.

The root level is mainly used to control the data coherence induced by the service publication in the system. There is only one GLWSM node. The leaf level has many GLWSM nodes. Each GLWSM leaf node is attached to a mobile access network. A GLWSM covers a geographical region that represents the location areas of the mobile access network with which it is associated. Two different GLWSM nodes cannot cover the same location areas. A SAS can be associated with many GLWSMs. A supplier's particular web service can be distributed over many SAS. The main characteristic of the leaf nodes is that they are dependent on the location areas of the network operator to ensure the mobility tracking, to maintain service execution in a SAS, and to coordinate the service migration when a mobile client moves to location areas controlled by another GLWSM node where the service in execution exists. Otherwise, the service execution will continue to be provided by the SAS in progress. We impose a migration delay constraint of 2 seconds to migrate a service.

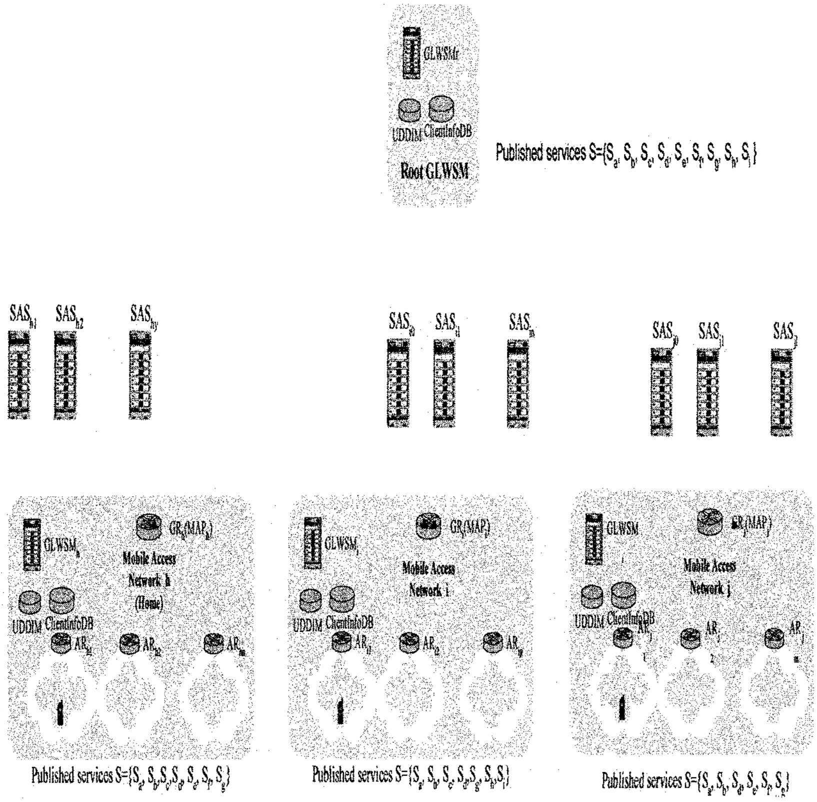


Figure 2. GLWSA topology

The service publication is controlled by the GLWSM root node and others functionalities (authentication and authorization, subscription, tracking mobile position, coordination of the migration, lookup) are offered by the leaf nodes. These are the relation properties of GWLSA:

Covered Areas: In the topology suggested, the location areas covered by two different GLWSM leaf nodes are disjoined.

Visibility Relation: Every GLWSM node that offers a service S_i knows all GLWSM nodes which offer the same service.

Home Node is the GLWSM leaf node where the mobile client is registered. Every mobile client is associated with a single home node. The mobile client used the home node to authenticate himself and to lookup for a service.

Visited Node is a GLWSM leaf node other than the home node of a mobile client.

Nearest Node is a GLWSM leaf node where the mobile client is located.

Nearest SAS is the SAS attached to the nearest GLWSM node and where the service requested by a mobile client is in execution phase.

4. GLWSA MATHEMATICAL FORMALISM

In order to facilitate the interactions implementation of the principal functionalities of the system with the databases UDDIM and "ClientInfosDB", we formalized the GLWSA concepts in an algebraic language.

4.1 Sets of discovering servers and location areas

Let E be the set of discovering servers GLWSM deployed at the leaf level of the topology tree. By definition:

$$E = \{GLWSM_1, GLWSM_2, ..., GLWSM_p\} \quad \text{(eq. 1)}$$

where p represents the cardinal of E or the number of GLWSM deployed in topology.

Let Z be the set of location areas covered by the GLWSA topology. By definition:

$$Z = \{Z_1, Z_2, ..., Z_t\} \quad \text{(eq. 2)}$$

where t represents the cardinal of Z or the number of the mobile network location areas associated the GLWSA topology.

Mobile access network: A mobile access network is a set of mobile network resources giving access to the network IP. The location areas covered by a mobile access network constitute a subset of Z which share the same MAP (Mobile Anchor Point).

Domain: We call domain a mobile access network controlled by a node GLWSM.

Adjacency: Two discovery servers $GLWSM_j$ and $GLWSM_{j+1}$ are contiguous if they have a common border of location areas.

Properties

Finished sets: At any moment, the sets E and Z are finished and determined sets. The cardinal of E or Z can increase or decrease dependently if there are an addition or a withdrawal of a GLWSM$_i$ node or a location area Z$_x$ in the GLWSA topology.

Topology control: The root node is the single discovering services server which manages (adds, withdraws, modifies) the topological structure deployed in system GLWSA.

Location areas covered by a leaf node GLWSM: Each leaf node GLWSM$_i$ is associated with a domain D$_i$. This association means that the node GLWSM$_i$ covers the location areas D$_i$.

Disjunction of covered location areas: Two distinct nodes GLWSM$_i$ and GLWSM$_j$ cover disjoined location areas. Thus, if ZL$_i$ and ZL$_j$ represent the location areas covered respectively by GLWSM$_i$ and GLWSM$_j$, then:

$$ZL_i \cap ZL_j = \phi \qquad \text{(eq.3)}$$

4.2 Sets of deployed services and application servers in GLWSA topology

Let F be a set of supplier applications servers SAS deployed in the GLWSA topology. By definition:

$$F = \{SAS_1, SAS_2, ..., SAS_m\} \qquad \text{(eq. 4)}$$

where m represents the cardinal of F or the number of SAS deployed in topology.

Let G be the set of Web services deployed in the GLWSA topology. By definition :

$$G = \{S_1, S_2, ..., S_z\} \qquad \text{(eq.5)}$$

where z represents the cardinal of G or the number of deployed Web services in the topology.

Definitions

Services vector: We call a services vector $\overrightarrow{V_k}$, the set of deployed services deployed at GLWSMk node. We have:

$$\overrightarrow{V_k} = [S_a S_b .....S_g] \bullet \overrightarrow{GLWSM_k} \qquad \text{(eq. 6)}$$

where $\{S_a, S_b, ..., S_g\}$ are a subset of G.

service presence: A service Web S_j is present at GLWSMk node if and only if S_j is provided by at least SAS associated with GLWSMk. Consequently, if Sj is present y time at GLWSMk node, then the measurement of presence p is:

$$p = y \qquad \text{(eq.7)}$$

Vector of service presence: We call vector of service presence $\overrightarrow{PV_k}$, the vector materializing the set of web services present at the GLWSMk node.

$$\overrightarrow{PV_k} = [p_{1k}\, p_{2k} \ldots\ldots p_{zk}] \bullet \overrightarrow{GLWSM_k} \qquad \text{(eq. 8)}$$

where p_{1k}, p_{2k},..., p_{zk} represent the presence of web services S_1, S_2,..., S_z of G at $GLWSM_k$ node. Matrix of service presence: We call matrix of service presence, the matrix M allowing to identify the presence of web services to any node GLWSM in the GLWSA topology. By definition:

$$M = \begin{bmatrix} p_{11} & p_{12} & p_{1p} \\ p_{21} & p_{22} & p_{2p} \\ & & \\ p_{z1} & p_{z2} & p_{zp} \end{bmatrix} \qquad \text{(eq. 9)}$$

where p_i represents the presence of the service S_i at $GLWSM_k$ node.

Properties

Finished sets: At any moment, the sets F and G are finished and determined sets. The cardinal of F or G can increase or decrease, dependently if there are an addition or a withdrawal of a supplier application server SAS_i or a service Web S_x in the GLWSA topology.

Number of offered services: A supplier application can offer one or more web services Web S_i.

Root node root and matrix of service presence: At any moment, the root node $GLWSM_r$ entirely knows the matrix of service presence M.

Relations

Supplier application servers and discovering services servers: Each application server SAS_k is associated with a discovering services server (leaf node) $GLWSM_i$ which covers its geographical position, and possibly other $GLWSM_c$ contiguous to $GLWSM_i$, if its distance compared to the associated discovering service nodes is lower or equal to the maximum distance tolerated between SAS and an associated node GLWSM of theGLWSA topology.

$$D(SAS_k, GLWSM_i) \leq D_{\max} \qquad \text{(eq. 10)}$$

where $D(SAS_k, GLWSM_i)$ is the distance between SAS_k and $GLWSM_i$.

4.3 Service publication

The publication of a service Web S_{z+1} in system GLWSA implies three essential operations:

1. unify the set G and the singleton { S_{z+1} }, we have:

$$G = G \cup \{S_{z+1}\} = \{S_1, S_2, ..., S_z, S_{z+1}\} \qquad \text{(eq. 11)}$$

on the level of the data base UDDIM, this operation consists of adding to the root node $GLWSM_h$ a new record to the service and agreement of service tables.

2. To add a new row L_{z+1} to the matrix of presence M, we have:

$$M = \begin{bmatrix} p_{11} & p_{12} & p_{1p} \\ p_{21} & p_{22} & p_{2p} \\ & & \\ p_{(z+1)1} & p_{(z+1)2} & p_{(z+1)p} \end{bmatrix} \qquad \text{(eq.12)}$$

on the level of the data base UDDIM, this operation consists of adding to any leaf node $GLWSM_i$ that has a non null presence of service $p_{z+1,i}$ a new record to the tables representing the service S_{z+1}, the agreement of service and the agreement of service at the particular $GLWSM_i$.

3. For any $GLWSM_i$ node having a non null presence of service $p_{z+1,i}$, confirm the recording of the service information published to the root node $GLWSM_h$.

4.4 Lookup service

The lookup of a web service S_d at the home node $GLWSM_h$ of a mobile customer CM implies six essential operations:

1. To locate the mobile client CM by formulating a request of position to the location server LCS.

2. To determine the server $GLWSM_s$ that covers the current geographical position POS_c of the mobile customer. That is equivalent in database algebraic language to select in the GLWSM nodes table, the node $GLWSM_s$ which covers the current geographical position POS_c of the client CM. This operation is equivalent to the following instruction:

$$\{T[GLWSM] \;\; where \;\; POS_c \subset AREA\} \quad [GLWSM_s] \qquad \text{(eq.13)}$$

where *AREA* materializes the geographical area covered by the node $GLWSM_s$.

3. To check in UDDIM_h (associate with the home node GLWSM_h) if the presence of the S_d service is non null at the node $GLWSM_s$. From the point of view of the database, this operation consists in making a projection on the GLWSM node axis of the *table T[S,GLWSM,ADDRESS]* identifying the

agreement of service to a node when the service has S_d as value and GLWSM has $GLWSM_s$ as value. This operation is equivalent to the following instruction:

$\{T[S, GLWSM, ADDRESS]\ \ where\ \ S = S_d \wedge GLWSM = GLWSM_s\}\ \ [GLWSM]$ (eq.14)

4. To select all application servers SAS associated with $GLWSM_s$ and that offer the concerned service S_d. This operation consists in making a projection on the address axis of the table *T[S, GLWSM, ADDRESS]* identifying the agreement of service to a node when the service has S_d as value and GLWSM has $GLWSM_s$ as value. This operation is equivalent to the following instruction:

$\{T[S, GLWSM, ADDRESS]\ \ where\ \ S = S_d \wedge GLWSM = GLWSM_s\}\ \ [ADDRESS]$ (eq.15)

5. To check if the QoS criteria meet the QoS level required by the mobile client CM; this phase is optional if the mobile client did not require QoS in his service discovering request.

6. To return to mobile client CM, the URL address of the service S_d that meets the QoS level.

4.5 Coordination of the service migration

The coordination of migration of a web service S_d that is in execution to a current application server SAS_c associated with the current node $GLWSM_c$, begins when the node $GLWSM_c$ is notified by the location server LCS that the mobile client CM (implied in the execution of the S_d service) moves out the $GLWSM_c$ covered area. The following steps are executed thereafter:

1. To determine the next node $GLWSM_n$ that covers the current geographical position POS_c of the mobile client CM. This operation consists in checking which GLWSM nodes of the set E covers the current geographical position of the mobile client CM. This operation is equivalent in the database algebraic language to the instruction eq.13.

2. To check the existence of service S_d and to select all application servers SAS associated with $GLWSM_n$ and that offer the service S_d. This operation is equivalent in the database algebraic language to the instruction eq.15.

3. To check if the QoS criteria meet the QoS level required by the mobile client CM; this phase is optional if the mobile client did not require QoS in his service discovering request.

4. To select the next application server SAS_n where the service migration will be carried out.

5. The node $GLWSM_c$ notifies to the SAS_c about the SAS_n URL address where migration of the service S_d migration must take off.

6. The application server SAS_c notifies to the $GLWMS_c$ of the end service S_d migration at SAS_n server.

7. The *$GLWMS_c$* sends to the *$GLWMS_n$* a message to continue the position tracking of the mobile client CM in its covered area.

5. IMPLEMENTATION DETAILS AND EVALUATION

To evaluate the proposed GLWSA architecture, we built a prototype using the Java programming language (Jbuider 7 and Sun Message Queue 3.5). This prototype implements the functionalities of the GLWSM and communicates with the LCS and the SAS servers. Communication with the LCS was carried out through Ericsson MPS 6.0 emulator. The emulator implements the MLP V3.0 protocol to determine the geocraphical location of a client (or a group of clients) moving over the network topology. We generate a network topology with the network density set to suburban, the distance between two base stations is set to 5000 meters and created a mobile trajectory with a constant speed value.

In Figure 3, the $GLWSM_h$ is the home GLWSM domain of the target mobile client and its geographical covered areas, the $GLWSM_1$ and $GLWSM_2$ are the visited GLWSM domains of the target mobile client with their associated geographical covered areas. In our tests, we used three constant speeds 50 km/h, 100 km/h and 200 km/h. By using the configuration settings described above, a static route file that contains the current cell identifier where the mobile resides, the relative distance between the mobile and the current base station and the mobile position data calculated each 10 seconds and given in geographical system coordinate (latitude/longitude/altitude) is created. The MPS 6.0 emulator calculates the client position between two consecutive offset times of the route file (for example between 0 and 10 seconds) by using the interpolation operations. But the formula used to do the interpolation operations is not given by the MPS tool specifications. At the beginning of the simulation, the emulator starts a clock and reads the mobile position in function clock time in the static route file created. We used the database management system Oracle 9i to store data in UDDIM registry and ClientInfosDB. We used JUDDI [8] and appended UDDIM API to interact with the registry UDDIM. The machines used to materialize the GLWSM, the SAS and the LCS are similar (1.2 GHz Pentium III with 512 Mo RAM). The machines are connected in a wireless LAN. The WLAN has a transmision rate of 11 Mbits/s and is compliant IEEE 802.11b.

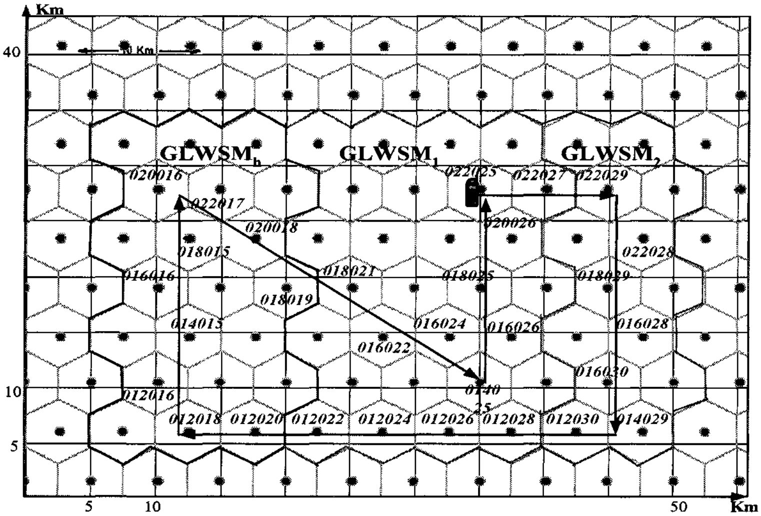


Figure 3. The generated network topology and trajectory of a target mobile client.

We define the round trip time (RTT) as the time difference between the reception time of the response at a client machine and the time when the client sent a request to a server. In Figure 4, we measured the RTT of a nearest service lookup without QoS by supposing that a mobile client sends a request each 2 seconds to his home GLWSM. The nearest lookup service request parameters sent are: service identifier and mobile identifer. The total size of sent parameters is 80 bytes.

We found a minimal RTT value of 20 milliseconds, an average RTT of 28 milliseconds and a maximum RTT of 52 milliseconds. The measure dispersion is 14.16 milliseconds.

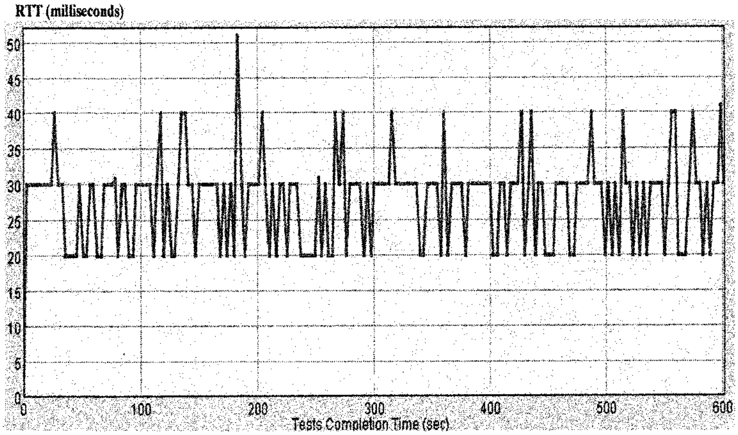


Figure 4. Round Trip Time of a nearest service lookup without QoS in case of one client.

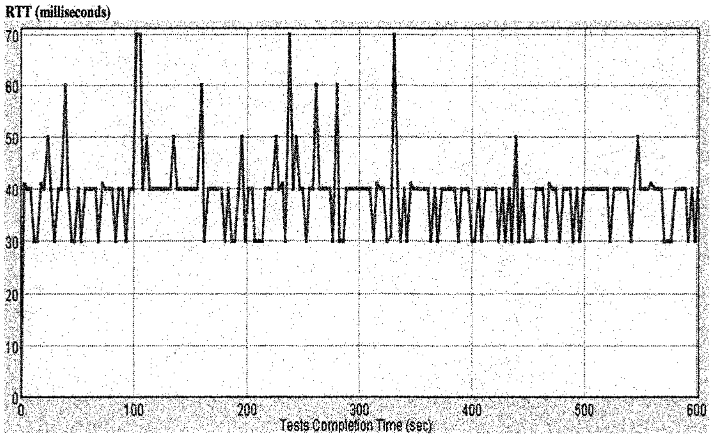


Figure 5. Round Trip Time of a nearest service lookup with QoS in case of one client.

To measure the RTT of the nearest service lookup with QoS, we suppose that a client sent to its home GLWSM a nearest lookup service request containing parameters: service identifier, mobile identifer, required service cost, required bandwidth and required SAS factor utilization. The total size of sent parameters is 104 bytes. We found a minimal RTT value of 30 milliseconds, an average RTT of 41 milliseconds and a maximum RTT of 70 milliseconds (Figure 5). The measure dispersion is 15.57 milliseconds.

The coordination time of a service migration shown in Figure 6 represents the RTT to send the URL of the next SAS (that implement the service in execution of a mobile client) and the mobile identifier parameter of a mobile client (who just changed the SAS domain) to the current SAS in execution. At the receiving of the message, the current SAS just sends back an acknowledgement to the GLWSM sender. Then, the GLWSM sender notifies the next GLWSM to track the target mobile. The coordination time of a service migration has an average RTT of 11 milliseconds, a minimal value of 7 milliseconds and a peak value of 40 milliseconds. . The measure dispersion is 5.80 milliseconds. We varied the speed of the target mobile and we remarked that the speed does not have a direct impact in the coordination of the service migration.

Coordination test of the service migration with QoS (Figure 7) consists first to send the URL address of the next SAS and the mobile identifier to the current SAS if a mobile client moves in the area covered by the next SAS. At the reception of the message, the current SAS just sends back an acknowledgement to the GLWSM sender (current GLWSM). Then, upon receiving the receipt of service migration end, the current GLWSM notifies the next GLWSM to track the position of the mobile client who just enter into its covered area. To track the mobile client, the current GLWSM sends to the next GLWSM the relevant information to do this operation (mobile identifier, service identifier, next SAS URL, client required service cost, client required bandwidth, client required SAS utilization rate). The total size of information sent to the next GLWSM is 104 bytes. The coordination time of a service migration has an average RTT of 13 milliseconds, a minimal value of 8 milliseconds and a peak value of 65 milliseconds. We varied the speed of the target mobile and we remarked that the speed does not have a direct impact in the coordination of the service migration. The measure dispersion is 6.47 milliseconds. Compare to the service migration without QoS, we found a mean RTT variation of 2 milliseconds. This variation is due principally to the increasing of 24 bytes size parameters sent to the current SAS.

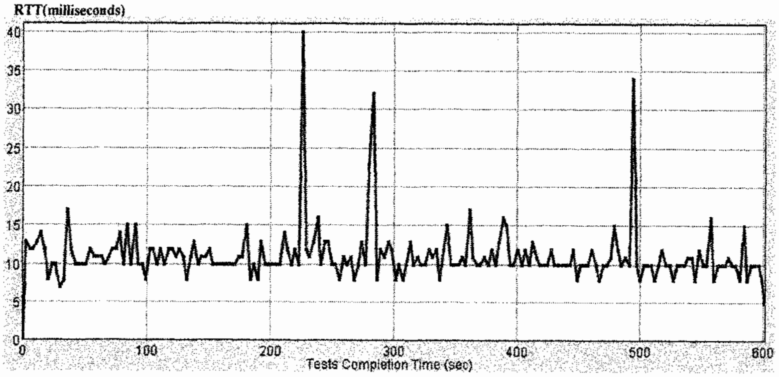


Figure 6. Coordination time of a service migration without QoS.

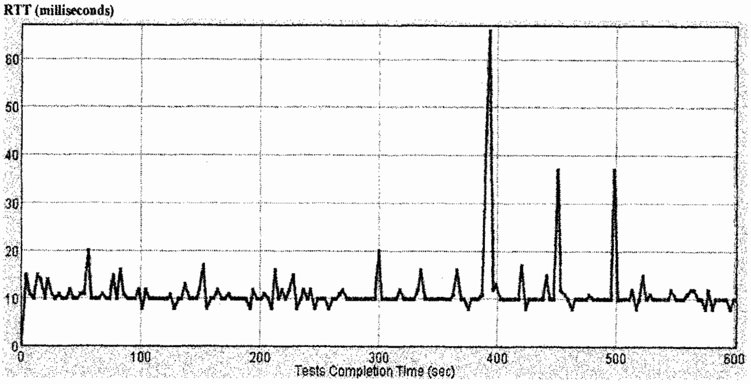


Figure 7. Coordination time of a service migration with QoS.

Meanwhile, as we imposed that the delay migration constraint must be less than or equals to 2 seconds, if a mobile client has a speed of 300Km/h when the migration is relevated, the target mobile will be at 166,67 meters of the precedent GLWSM domain when the service migration will be terminated. As the service migration for SAS to SAS has a maximum average rate of 300 milliseconds [4], we will have a maximum total service migration time (SAS service migration time and coordination migration time) of 355 milliseconds which is less than the delay migration service constraint. Thus, the system has a margin time 1645 milliseconds for huge applications.

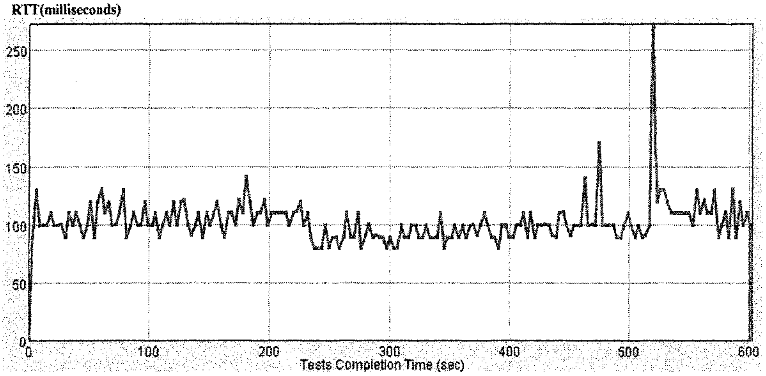


Figure 8. Geo-located web service publication time in two GLWSM leaf nodes.

Figure 8 shows a geo-located web service publication time in two GLWSM leaf nodes. To measure the service publication, we suppose that an authorized supplier sent a size of 1116 bytes parameter data (service, agreement service, agreement node entities) to publish to the root GLWSM machine. Then, the root GLWSM stores the data in his UDDIM and forwards them to the publication queue listened by the two GLWSM leaf nodes. After storing the forwarded data in their UDDIM, each GLWSM leaf node sends back a recept to the root GLWSM. We found an average RTT of 105 milliseconds with a minimal and maximal RTT of 88 and 300 milliseconds during 10 minutes observation. We also analyzed the publication time in function of the number GLWSM leaf nodes (which increases the size of parameters sent in a publication process) where they are published. We found that the publication time increases when the number of GLWSM leaf nodes increases too. We found an average RTT of 110, 147 and 247 milliseconds for 5, 10 and 20 GLWSM leaf nodes, respectively.

6. CONCLUSION

We presented in this paper a discovering architecture for geo-located web services for the next generation mobile networks. With tis architecture, mobile clients can discover geo-located web services and maintain the service execution closest to their location context. We extended the UDDI registry and MLP protocol to reach this goal. Tests executed in the GLWSA system show that the system gives good response in lookup, migration and publication service in the context of a client mobility. Future work will present the mechanism of QoS collection, the thematic location and will consider the adaption of data presentation in different client format machines (xHTML *eXtensible HTML*, cHTML *compact HTML*, WML (*Wireless Markup Language*) and the conviviality of the prototype.

REFERENCES

1. **3GPP**, *Functional stage 2 description of location services in UMTS*, Technical Specifications TS 23.171 V3.9.0, Reference http://www.arib.or.jp/IMT-2000/ARIB-spec/ARIB/23171_300.PDF, September 2002.
2. **G. Ballintijn, A. S. Tanenbaum** and **M. R. Van Steen**, *Locating Objects in a Wide-area System*, PhD Thesis, Amsterdam University, Globe Project, Reference http://www.cs.vu.nl/res/theses/ballintijn_thesis.pdf, 2003.
3. **M. Barbeau**, *Bandwidth Usage Analysis of Service Location Protocol*, Workshop on Pervasive Computing, International Conference on Parallel Processing, Toronto, Reference http://citeseer.nj.nec.com/barbeau00bandwidth.html, August 2000.
4. **S. Bouchenak, D. Hagimont, S. Krakowiak, N. De Palma** and **F. Boyer**. *Experiences Implementing Efficient Java Thread Serialization, Mobility and Persistence*, INRIA Technical Report No. RR-4662, December 2002.
5. **M.A Dru** and **S. Saada**, *Location based mobile services: the essentials*, Alcatel Telecommunications review, pp. 71-76, 1st quarter 2001.
6. **E. Guttman**, *Service Location Protocol : Automatic Discovery of IP Network Services*, IEEE Internet Computing, pp. 71-80, July-August 1999.
7. **N. Harashima, T. Okoshi, J. Nakazawa, Y. Tobe**, and **H. Tokuda**, *AMRB: Toward Location Migration Transparency of Services*, IEEE International Conference on Parallel and Distributed Systems, pp. 305-314, ICPADS 2001.
8. **jUDDI**, http://ws.apache.org/juddi/ .
9. **Location Inter-operability Forum (LIF)**, *Mobile Location Protocol*, LIF TS 101 Specification, Version 3.0.0 6, Reference http://dan.greening.name/profession/manuscripts/LIF%20TS%20101%20v3.0.0.pdf, June 2002.
10. **F. Michahelles, M. Samulowitz** and **B. Schiele**, *Detecting Context in Distributed Sensor Networks by Using Smart Context-Aware Packets.* In International Conference on Architecture of Computing Systems, Reference http://www.vision.*ethz.ch/publ/arcs02.html, ARCS 2002.*
11. **S. Panagiotakis** and **A. ALonistioti**, *Intelligent Service Mediation, for supporting advanced location and mobility-aware service provisioning in reconfigurable mobile networks*, IEEE Wireless Communications, Vol. 9, pp. 28-38 , October 2002.
12. **A. Y. Seydim, M. H. Dunham,** and **V. Kumar**, *An architecture for location dependent query processing*, Proceedings in 12th International Conference on Database and Expert System Applications , pp. 549-555, DEXA 2001.
13. **UDDI**, http://www.uddi.org/specification.html.

Authors Index