

UNIVERSITY OF OSLO  
Department of Informatics

Service Orchestration  
in IMS Centred  
Architecture

Master Thesis

Arlene Marie Pearce

August 2008





## Abstract

Current trends in communications technology favour modular, reusable service components delivered over converged networks. This service oriented approach gives rise to a new breed of composite, personalised services, but so far web-based providers dominate this market. In order to adapt and compete, telecom operators need to implement effective methods for developing, integrating and delivering services.

IP Multimedia Subsystem (IMS) is a fairly new framework being tested and rolled-out by telecom operators around the globe. It has a layered architecture that accommodates the delivery of personalised, composite, multimedia services over converged heterogeneous networks. However, the static nature of IMS service chaining limits its flexibility.

This thesis proposes a model for dynamic service orchestration in IMS centred architecture. The proposal addresses orchestration of typical IMS services running on native SIP application servers as well as the incorporation of a variety of services residing in foreign domains. In particular, the possibility to include external Web Services in composite services the IMS domain is examined. A new hierarchical configuration of service brokers is introduced and a basis prototype is implemented in the scenario of a composite Presence service.

The proposed model in this thesis augments the current IMS service provisioning mechanism in several ways: It introduces the notion of dynamic service brokering. It adds explicit support for non-telecom services in native IMS application servers. Furthermore, the proposed model utilises only reference points already present in the existing IMS specification, no additional protocols or control functions were needed. The functionality introduced is meant to improve the flexibility of IMS service provisioning in terms of both the type of services that can be offered natively as well as the types of services that can be supported from third parties.

The intention of this thesis is to provide a model for IMS service orchestration. In particular, it identifies the technologies that make such an architecture feasible as well as points out best practices for maintaining performance levels. The model was verified by implementing a prototype that blends native IMS services and external Web Services in the IMS domain.

## Preface

This work was carried out in the period September 2007 to August 2008 as a partial requirement for the degree Master of Science in Informatics at the Department of Informatics of the University of Oslo in Norway.

The Research was carried in cooperation with Telenor Research and Innovation (R&I), in part within the framework of the EU project Mobicome. The main thesis advisor was Terje Jensen of Telenor R&I and Norwegian University of Science and Technology (NTNU). The secondary thesis advisor was Carsten Griwodz of Simula Research Laboratories and the University of Oslo.

In the course of this research, the following three papers were co-authored with Terje Jensen and Judith Rossebø:

- *On Understanding Availability of Services Based on IP Multimedia Subsystem*, to appear in the proceedings of the 18th International Teletraffic Congress (ITC) Specialist Seminar on Quality of Experience in Karlskrona Sweden, May 29, 2008. (presented by myself)
- *Combining IMS and Web 2.0 - Understanding the Availability*, accepted to the International Conference on Intelligence in Networks (ICIN) 2008 in Bordeaux France, will be presented October 23, 2008. (presented by myself)
- *What is Availability?* to be published in *Telektronikk* in 2009.

This thesis is not intended to give a comprehensive introduction to the protocols and interfaces used in IMS. Chapter 2 does provide an overview of IMS. For those who would like to learn more, the book *The IMS: IP Multimedia Concepts and Services* by Poikselka et. al is recommended as a starting point. Specifications referenced in this thesis are listed in the bibliography. Many acronyms are used throughout the thesis. An alphabetic list of acronyms and what they stand for can be found in appendix A. The prototype code and Javadoc is appended on a CDROM.

I would like to thank my supervisors Terje Jensen and Carsten Griwodz for their good advice, their patience and their humour. I would also like to thank colleagues and friends at Telenor R&I and the Mobicome team for providing me with a great working environment and great opportunities. Last but not least I would like to thank my husband Andre for constant moral support, for proof reading at 2 AM and for helping me to figure out the meaning behind bizzare Latex error messages. :)

*Arlene Pearce*

*Oslo, August 2008*

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	2
1.2	Background . . . . .	3
1.3	Objectives . . . . .	3
1.4	Assumptions and Limitations . . . . .	4
1.5	Contributions . . . . .	5
1.6	Outline . . . . .	5
<b>2</b>	<b>IP Multimedia Subsystems</b>	<b>8</b>
2.1	Overview . . . . .	8
2.2	Identities in IMS . . . . .	9
2.2.1	Private User Identity . . . . .	9
2.2.2	Public User Identity . . . . .	9
2.2.3	Public Service Identity . . . . .	11
2.2.4	Service Profile . . . . .	11
2.3	Databases . . . . .	11
2.3.1	Home Subscriber Server . . . . .	12
2.3.2	S-CSCF Media Policy Database . . . . .	12
2.3.3	XML Document Management Server . . . . .	12
2.4	Serving - Call Session Control Function . . . . .	12
2.4.1	Initial Filter Criteria . . . . .	13
2.5	SIP Application Server . . . . .	13
2.6	Service Capability Interaction Manager . . . . .	14
<b>3</b>	<b>Related Technologies</b>	<b>16</b>
3.1	Service Oriented Architecture . . . . .	16
3.2	User Oriented Architecture . . . . .	17
3.3	Event Driven Architecture . . . . .	18
3.4	Web Services . . . . .	19
3.4.1	Extensible Markup Language (XML) . . . . .	20
3.4.2	Web Services Description Language (WSDL) . . . . .	21
3.4.3	Universal Description Discovery and Integration (UDDI) . . . . .	22
3.4.4	SOAP . . . . .	22
3.4.5	Web Services Invocation Framework . . . . .	23
3.4.6	Representation State Transfer (REST) . . . . .	23
3.4.7	OSA Parlay X . . . . .	24

3.4.8	Web Services Security . . . . .	26
3.5	Service Orchestration . . . . .	26
3.5.1	Choreography vs Orchestration . . . . .	26
3.5.2	Limitations . . . . .	27
3.6	Enterprise Service Bus . . . . .	27
3.7	Service Delivery Platforms . . . . .	28
3.7.1	Telephony Application Programming Interface (TAPI) . . . . .	28
3.7.2	SIP Common Gateway Interface(SIP CGI) . . . . .	28
3.7.3	Service Level Execution Environment . . . . .	29
3.7.4	SIP Servlets . . . . .	29
3.7.5	SDPs and IMS Requirements . . . . .	30
3.8	Operational Support System . . . . .	30
3.9	Web 2.0 Principles . . . . .	31
3.9.1	General Characteristics . . . . .	31
3.9.2	Mashups . . . . .	33
<b>4</b>	<b>Problem Formulation</b> . . . . .	<b>34</b>
4.1	General Challenges in Distributed Systems . . . . .	34
4.2	IMS Requirements for Service Orchestration . . . . .	35
4.3	The SCIM as a Service Broker . . . . .	35
4.3.1	Dynamic Service Brokering . . . . .	36
4.3.2	Logical Placement . . . . .	36
4.3.3	Constellation: Centralised, Distributed or Hybrid? . . . . .	37
4.4	Orchestration Options . . . . .	39
4.5	Quality of Service (QoS) Aspects . . . . .	39
4.5.1	Availability . . . . .	40
4.5.2	Session Setup Latency . . . . .	40
4.5.3	SIP Proxy Latency . . . . .	41
4.5.4	Web Services Latency . . . . .	42
4.6	Problem Summary . . . . .	43
4.7	Hypothesis . . . . .	44
<b>5</b>	<b>Model</b> . . . . .	<b>46</b>
5.1	Existing Service Creation Interfaces . . . . .	46
5.2	IMS Service Routing . . . . .	47
5.3	Service Provisioning and Request Routing . . . . .	49
5.4	Incorporating Web Services . . . . .	51
5.5	SIP Service Domain . . . . .	52
5.6	SSD Service Broker Architecture . . . . .	52
5.6.1	Hierarchical Service Brokers . . . . .	53
5.6.2	SB Proxy or SB Back-to-Back UA . . . . .	54
5.7	Choice of Service Delivery Platform . . . . .	54
5.7.1	Mobicents Implementation of JSLEE . . . . .	54
5.7.2	Sailfin's Implementation of SIP Servlets . . . . .	55

<b>6</b>	<b>Implementation Details</b>	<b>56</b>
6.1	Scenario . . . . .	56
6.1.1	Default Service Profile . . . . .	56
6.1.2	Home Service Profile . . . . .	56
6.1.3	Services . . . . .	57
6.1.4	Use Cases . . . . .	57
6.2	IMS Core Component . . . . .	57
6.3	Client Implementation . . . . .	59
6.4	SSD System Model . . . . .	59
6.4.1	SSD Components and Deployment . . . . .	59
6.4.2	Presence Service . . . . .	60
<b>7</b>	<b>Experiment and Results</b>	<b>63</b>
7.1	Availability . . . . .	63
7.2	Latency . . . . .	63
7.2.1	Response Times for Call-Setup . . . . .	64
7.2.2	Response Times for SIP Presence . . . . .	65
7.2.3	Response Time for Web 2.0 Presence . . . . .	66
<b>8</b>	<b>Conclusion</b>	<b>68</b>
8.0.4	IMS Latencies . . . . .	68
8.0.5	JSLEE from a Development Environment . . . . .	68
8.0.6	SIP Servlets from a Development Perspective . . . . .	69
8.1	Summary . . . . .	70
<b>9</b>	<b>Further Work</b>	<b>72</b>
9.1	Verification of Measurements and Further Development . . . . .	72
9.2	Security . . . . .	72
9.3	Database Solutions . . . . .	73
9.3.1	Standardisation of XDMS usage in IMS . . . . .	73
9.3.2	Generic User Profile . . . . .	73
9.3.3	Lightweight Directory Access Protocol . . . . .	74
	<b>Appendix A – Acronyms</b>	<b>78</b>
	<b>Appendix B – ITCss Paper</b>	<b>80</b>
	<b>Appendix C – ICIN Paper</b>	<b>90</b>

# List of Figures

1.1	Silo Architecture vs Layered Architecture . . . . .	3
1.2	Thesis Overview . . . . .	7
2.1	IMS Architectural Overview . . . . .	9
2.2	Identities in IMS . . . . .	10
2.3	SIP Application Server . . . . .	14
3.1	Relationship between the Enabling Technologies . . . . .	17
3.2	SIP INVITE Client Transaction - state machine . . . . .	19
3.3	Standard Web Services Interactions - Overview . . . . .	20
3.4	Typical Web Services Protocol Stack [1] . . . . .	21
3.5	Choreography vs Orchestration . . . . .	27
3.6	Choreography and Orchestration . . . . .	28
3.7	Overview of JSLEE Architecture [2] . . . . .	29
3.8	NGOSS . . . . .	30
4.1	SCIM: Centred vs. Distributed . . . . .	38
4.2	Hybrid SCIM . . . . .	38
4.3	Orchestration Levels . . . . .	40
4.4	Web Services Throughput: SOAP vs REST [3] . . . . .	44
5.1	Service Creation Interfaces . . . . .	47
5.2	Relationship between User and Service Profiles . . . . .	47
5.3	Relationship between iFC and SPT . . . . .	48
5.4	AS Invocation . . . . .	48
5.5	Generic Request Routing . . . . .	50
5.6	WS/IMS Scenario: Control in Foreign Domain . . . . .	51
5.7	Control in IMS Home Domain . . . . .	52
5.8	SSD with Hierarchical Service Brokers . . . . .	53
5.9	Mobicents . . . . .	55
6.1	Alice's Home Service Profile . . . . .	57
6.2	Call Established . . . . .	58
6.4	Fokus Open IMS Core [4] . . . . .	58
6.3	Use Cases . . . . .	59
6.5	SSD (and surroundings) Component Diagram . . . . .	60
6.6	OMA Presence SIMPLE (wikipedia.org) . . . . .	60
6.7	SSD (and surroundings) Component Diagram . . . . .	61

6.8	Composite Presence in SSD Sequence Diagram . . . . .	62
7.1	Trigger Points for Presence Service . . . . .	64
7.2	SIP Traffic for Alice's call to Bob . . . . .	65
7.3	SUBSCRIBE datagram . . . . .	66
7.4	Get WS Buddies . . . . .	67
8.1	SDP in Invite Message . . . . .	69

# List of Tables

3.1	SOAP vs. REST . . . . .	24
3.2	Overview of SDP Features (adapted from [5]) . . . . .	30
4.1	SIP Transmission Timers . . . . .	43
7.1	INVITE Delay: Bob as originator . . . . .	65
7.2	Response Time: SUBSCRIBE/PUBLISH . . . . .	65
7.3	Response Time for External Buddy Lists . . . . .	66
8.1	Similarities between HTTP Servlets and SIP Servlets . . . . .	70

# Chapter 1

## Introduction

Modern day telecommunication can be defined as "the transmission of information, as words, sounds, or images, usually over great distances, in the form of electromagnetic signals, as by telegraph, telephone, radio, or television" [6]. From Morse code and telegraphs to wireless broadband and the Internet, the evolution of signalling technology has given rise to a complex range of communication services. Telecommunications companies are trying, in some cases unsuccessfully, to adapt to these changes by changing their business models.

According to Gartner [7], "more than half of global tier-one telecom carriers trying to establish new lines of business will fail through 2010". Gartner's dire prediction is given in the context of the so-called IP revolution. Traditionally, telecommunications operators (telcos) survived on providing conduits for voice, and especially in recent decades, data communication. However, signalling speed, processing speed, and data storage capacity has increased exponentially, causing a dramatic fall in the price of communication technologies. In parallel, telecom regulations have undergone radical changes worldwide.

This combination of legal and technological changes has opened up the telecom market to new competitors. It has also given rise to a new breed of advanced services and a new breed of discerning users. Today's service provider may be anyone from individuals on the Internet, to software companies like Microsoft, to search engine companies like Google to telcos like Telenor. The competition is fierce and arguably, the modern telco must innovate effectively or die. IP everywhere, composite services and re-usability are key innovations in telecom today and the fairly new and much discussed IP Multimedia Subsystem (IMS) architecture is a promising candidate for driving such innovation.

IMS provides a framework for provisioning of rich multimedia (voice, video, data) services over both fixed and mobile networks. Native IMS protocols are based on IETF specifications. The most significant being Session Initiation Protocol (SIP), IP and Diameter. Although IMS supports legacy and non-IP transport protocols the goal is all-over-IP scenario. Generally speaking, efficient creation and delivery of composite telecom services is challenging. This thesis specifically addresses the orchestration of composite telecom services in IP Multimedia Subsystem (IMS) centred architecture.

## 1.1 Motivation

The motivation for carrying out this study fit into three main categories of service oriented architecture, web service trade-offs and perceived quality of service. We expand on these motivational elements in the following list:

- **Potential Benefits of Service Oriented Architecture:**

Service Oriented Architecture (SOA) is a loosely coupled distributed systems architecture that is often implemented through use of Web Services (WS) interfacing. SOA is explained in greater detail in section 3.1. The SOA notion of service orchestration is most often used synonymously to mean "Web Services Orchestration". As with many terms in software engineering, the definition may vary. For this discussion, service orchestration is defined as the process of dynamically arranging independent service modules into composite services for delivery to end users on request.

By adopting an effective service orchestration method, telecommunications companies (telcos) can leverage SOA's modularity and well defined interfaces to develop services more quickly and cost effectively as well as to host a wide range of third party services and content. In so doing, telcos can provide an attractive collection of services to new and existing subscribers to the benefit of both parties. The reusable nature of such a modular system also implies that telcos would be better equipped to react rapidly to emerging service trends and be first adopters of exciting new services.

- **Web Services Trade-offs**

This thesis proposes to study IMS Service Orchestration that incorporates the use of WS. However, WS architecture is traditionally applied to business process models. Telecom technology on the other hand is traditionally based on real-time services. This implies that latency and throughput requirements are strict and of the highest priority in telco service provisioning.

In addition interaction between services deployed in a telco domain must be secure and transactionally reliable. In the current WS architecture [1], there are no built-in mechanisms to govern the security between collaborating WS applications. Neither are there any rules based systems to ensure reliable transactions. This implies that cross domain transactions in the IMS service layer would be especially vulnerable to low security implementations of third party web services. Stringent security agreements would be essential and different levels of service availability may be required.

- **Efficiency and Perceived Quality of Service:**

The specifications defined in IMS define an open systems architecture for sending multimedia traffic over fixed and mobile networks. An open systems architecture presents the opportunity for distinct service providers in a widely distributed system to converge in a composite telecom infrastructure. This paints a dramatically different picture from that of traditional heterogeneous telecom infrastructures. An efficient method for service orchestration is thus imperative if this architecture is to succeed.

The IMS service layer can host many different services, each catering to hundreds of thousands of customers. Each service may in turn comprise of several different modules and each module may stem from a different provider. How can related resources be more efficiently distributed and managed in terms of speed and cost? Can the current architecture be streamlined without compromising the Quality of Service (QoS) levels required for multimedia telecommunication? Today's solutions are neither consistent nor compatible. The negative consequences of this disparity include limited efficiency when updating services or when sharing data with trusted domains.

## 1.2 Background

The historical picture of modern telcos is one of monopolies with supply chains tightly controlled by vertical integration. Typically, each service was provided on top of a so called "silo" architecture, often with proprietary solutions. Enterprise level integration was a challenge and multiple implementations of common processes was standard as was multiple instances of the same data. As illustrated in figure 1.1(a) a PSTN voice service over PSTN control mechanisms over PSTN network infrastructure. Similarly UMTS mobile services over UMTS control mechanisms over UMTS network infrastructure and data and media services similarly over their own control and dissemination systems. As illustrated in figure 1.1(b), today's market is rapidly moving towards layered architecture of horizontal integration and separation of concerns. Current trends favour communication over IP, modular and reusable components and converged technologies.

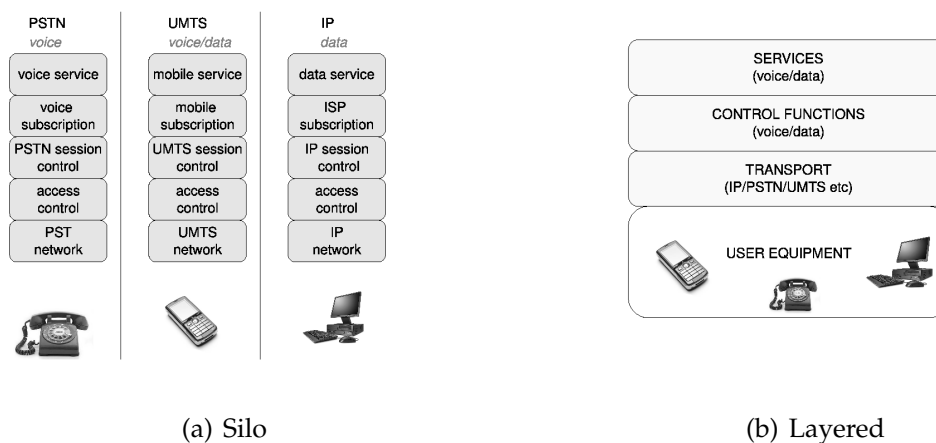


Figure 1.1: Silo Architecture vs Layered Architecture

## 1.3 Objectives

There are several perspectives from which to study this problem. In terms of the system's users, it can be approached from the perspective of the operators and/or service

providers or from the perspective of the end users. System functionality issues include signalling and streaming. This thesis approaches the issue from the perspective of the modern operator who is also in the process of adopting a second role as service provider.

The main objective is to design a feasible service layer architecture for orchestrating composite services in an IMS framework. This architecture should be able to incorporate web-based service modules from third party service providers and content providers.

In designing this architecture the feasibility of web services as a tool for service orchestration in IMS will be investigated. Of special interest is how web services will affect the latency of setting up end-to-end SIP sessions in an IMS service environment.

Therefore, the focus in this thesis is on how to take existing mechanisms in IMS and extend them with compatible technologies like Web Services in order to create a usable framework for dynamic service orchestration.

## 1.4 Assumptions and Limitations

The IMS architecture is extensive and research done on IMS so far is limited. Therefore, to make this undertaking feasible and focused in the time available we have imposed some practical limitations and assumptions as follows:

- **Preliminary Statistics**

The goal of this research is to design an architecture and verify its feasibility with a proof-of-concept prototype. The time will not permit for thorough statistical analysis so any measurements obtained will have to be verified in future work.

- **SIP Application Servers Only**

We limit our discussion to services housed on native SIP application Servers (SIP AS). SIP is one of the central protocols around which IMS architecture is built. It is the protocol on which basic, native IMS services are based and therefore the minimum requirement for the SIP AS.

For the sake of completeness we can mention that in addition to the SIP AS, two optional application servers are defined in the IMS service layer:

- IP Multimedia Services Switching Function (IM-SSF) - for the support of external CAMEL servers housing legacy services such as 2G, and
- Open Services Architecture Service Capability Servers (OSA SCS) - for the support of non-native services housed on external OSA servers.

These legacy and non-native services can add several layers of complexity to the question at hand and are thus more suited to a separate discussion. Furthermore, the goal of IMS is to have all services running over IP so the legacy support issue is less interesting.

- **Authentication and Security are Out of Scope**  
Authentication and Security offer their own research challenges in the IMS framework. In addition, SIP has inherent security vulnerabilities. To isolate our discussion from authentication issues we assume that all transactions are being carried out by registered IMS users. We will not have detailed discussions on security mechanisms, but we will bring it to the readers attention when security is a relevant issue that needs to be considered in future work.
- **Advanced Identity/Profile Management is Out of Scope**  
IMS offers advanced identity and profile management capabilities that are not considered here. A basic introduction to IMS identity management is discussed in section 2.2.
- **Databases Solutions are Not Covered in Depth**  
An in-depth discussion on IMS data management is also a research topic in itself. However, some IMS databases are briefly introduced in section 2.3 in the context of their role in service provisioning. To eliminate the case of database updates being performed for registered clients we assume that no user profile changes occur after registration.
- **Only Relevant IMS Entities are Discussed**  
There are many IMS entities in the control layer and in the transport layer that are not discussed here. These entities are either at a lower level of abstraction more relevant to media-specific discussions.
- **Presence Service Assumptions**  
For the SIP based Presence service we assume successful authorisations, no rejection of requests, and no failures.

## 1.5 Contributions

This thesis proposes architecture for dynamic service brokering in IMS centred architecture. The proposal addresses brokering of typical IMS services running on native SIP application servers as well as the incorporation of a variety of services residing in foreign domains. A new hierarchical configuration of service brokers is introduced and a basic scenario is demonstrated. This proposal is a possible augmentation to the current IMS service-brokering scenario, which only supports static service chaining of SIP services. Adding dynamism to IMS service composition would greatly improve the flexibility of the type of services that can be offered natively or supported from third parties.

## 1.6 Outline

The remainder of this thesis is arranged as follows:

- Chapter 2, IP Multimedia Subsystems, introduces the basic concepts of IMS with a focus on service composition mechanisms.

- Chapter 3, Relevant Technologies, presents related architectures and frameworks relevant to this discussion.
- Chapter 4, Problem Formulation, constructs a hypothesis on which to base a model.
- Chapter 5, Preliminary Model, provides a preliminary model.
- Chapter 6, Implementation Details, provides implementation details of the chosen solution.
- Chapter 7, Results, presents key observation from the implementation.
- Chapter 8, Conclusion, draws a conclusion based on the entire analysis.
- Chapter 9, Further Work, points out further research needed to build upon the work done in this thesis
- Appendix A, List of Acronyms, presents a list of of acronyms used in this Thesis.
- Appendix B, Co-authored Paper, On Understanding Availability of Services Based on IP Multimedia Subsystem.
- Appendix C, Co-authored Paper - Combining IMS and Web 2.0 - Understanding the Availability.

Figure 1.2 gives a graphical presentation of the thesis' general flow. It also illustrates the roles of the individual chapters and the relationship between them.

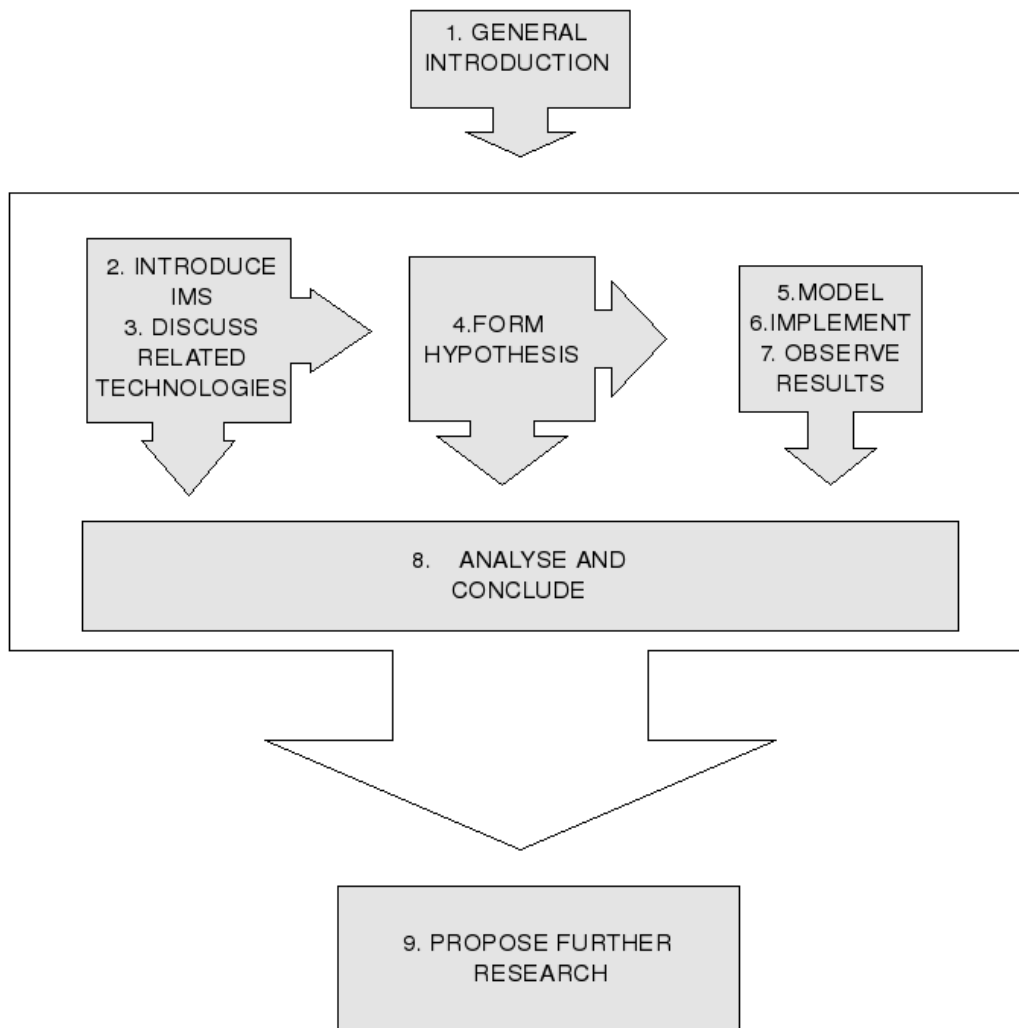


Figure 1.2: Thesis Overview

# Chapter 2

## IP Multimedia Subsystems

In this chapter, the basic concepts of IMS are introduced. The components involved in IMS service provisioning are focused upon. Section 2.1 gives an overview of IMS architecture with a focus on the key components involved in service creation. Section 2.2 presents the basic identity structure in IMS. Identities in IMS form an important part of the foundation of service brokering mechanisms in IMS. Section 2.3 gives a brief introduction to IMS databases with a focus on the Home Subscriber Server (HSS) which is the main IMS user database. Section 2.4 explains the role S-CSCF in IMS. The S-CSCF can be considered the element that provides the lowest level of service brokering in IMS. Section 2.5 explains the role of SIP application servers (SIP AS) in IMS. Finally, Section 2.6 discusses the concept of an IMS Service Capability Interaction Manager (SCIM).

### 2.1 Overview

IMS architecture is extensive with several multi-purpose components and optional roles. Here we will review the general mechanisms of the components that are central to the scope of this study, and, by extension, present the modules and interfaces necessary to complete the signal chain during service orchestration. See section 1.4 for a discussion on how the study scope was narrowed. All the presentations here are based on the 3GPP IMS specification [8].

As discussed in section 1.2, IMS is a layered architecture. It is generally considered to have three layers as depicted in figure 2.1. We will refer to these layers as the transport layer (transport services and media flows), the control layer (IMS signalling and session management) and an application layer (all deployed user services). Figure 2.1 presents an overview of these layers with a focus on the service layer and parts of the control layer that are directly relevant to service provisioning. In this simplified overview, it is the SIP Application Servers (SIP AS), the Serving-Call Session Control Function (S-CSCF), the Home Subscriber Server (HSS) and the User Equipment (UE) that are the most important components in our study. To provide a background for further analysis, sections 2.2 through 2.4 provide a more detailed introduction to each of these central components.

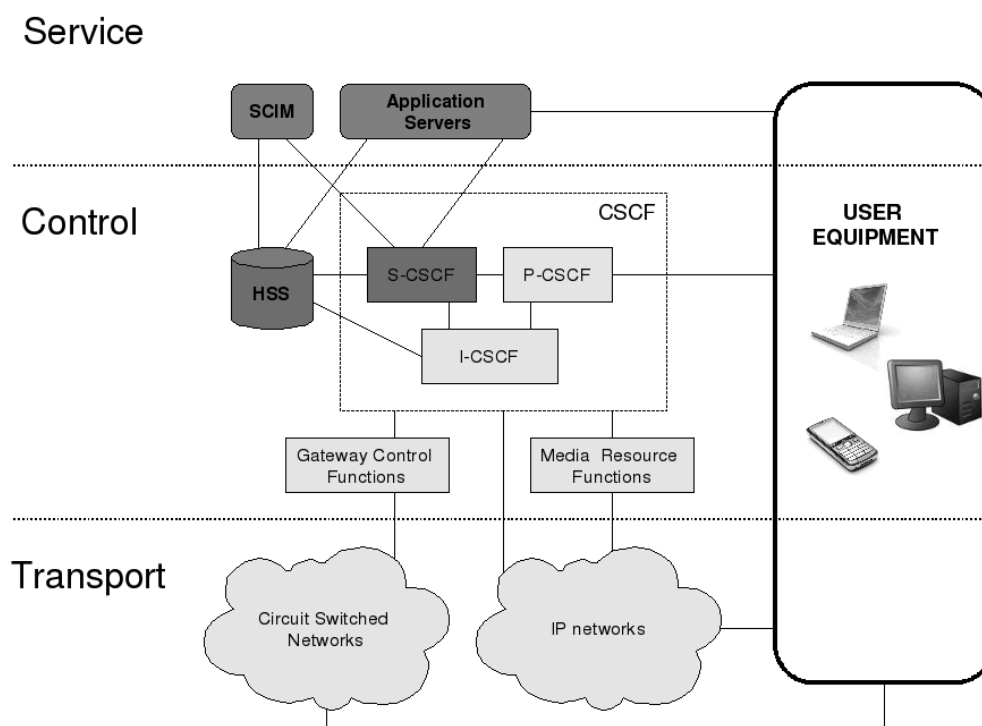


Figure 2.1: IMS Architectural Overview

## 2.2 Identities in IMS

Identity management in IMS plays an important role in service orchestration. Both users and services have unique identifiers that are used in the provisioning of composite services. We will focus on the role IMS identities play in personalised service composition. Greater detail on IMS identity management can be found in [8].

### 2.2.1 Private User Identity

Each IMS end user has a subscription in her home domain. The subscription is identified by at least one, permanently allocated, IMS Private User Identity (IMPI) which is a globally unique identifier. The IMPI is used to authenticate the user on registration and deregistration on the IMS network as well as for other authentication, authorisation and accounting related tasks (for example in charging procedures). It is however, not used for routing tasks; for this the IMS Public User Identity (IMPU) is used instead.

### 2.2.2 Public User Identity

An IMPI can in turn be associated with several IMPUs. IMPUs identify the origin or destination of SIP requests and responses and are therefore necessary for lookup and routing. They can take the form of a SIP URI<sup>1</sup> or a tel URI (telephone number)

<sup>1</sup>Uniform Resource Identifier

and they may be made publicly available (for example in a telephone catalogue). It is important to note that in traditional telephone systems, tel URIs typically identify devices, whereas in IMS they identify users. This difference is one of the pillars of the service oriented approach as applied telecom services.

However, there are several ways in which a user can force a telephone number or URI to be routed to one and only one of her devices. For example, an IMPU/UE combination can be uniquely identified with a Globally Routable User Agent URI (GRUU) [9]. Let us say that IMPU-A is associated with device-1, device-2 and device-3. A GRUU can be used to identify the combination IMPU-A/device-1. When needed, calls (SIP requests) addressed to IMPU-A can then be routed to device-1 only instead of being forked to all three devices associated with IMPU-A.

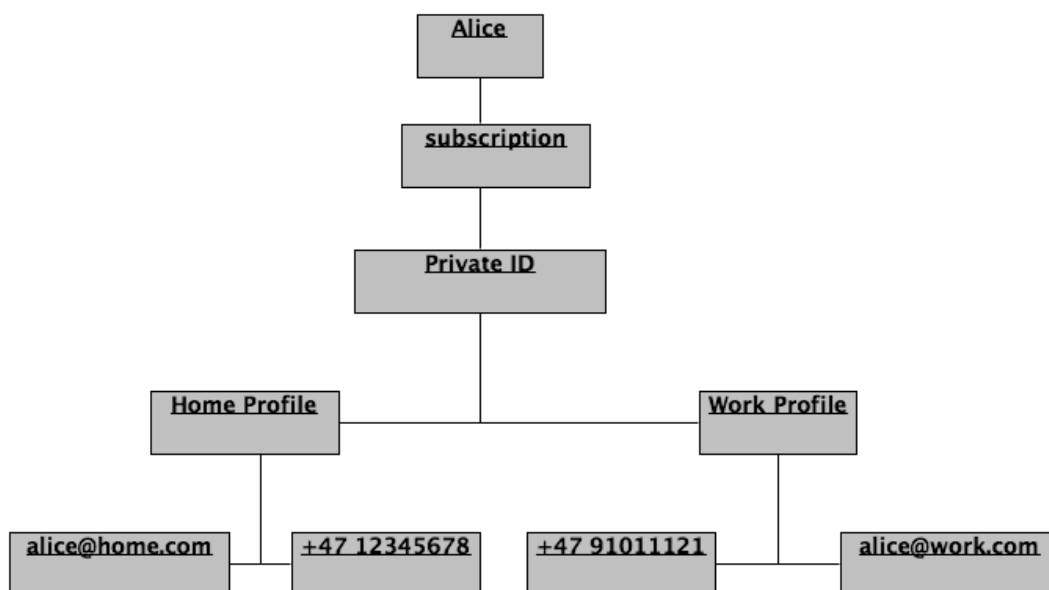


Figure 2.2: Identities in IMS

Figure 2.2 shows the relationship between one user and distinct instances of subscription, private and public identification and service profiles. It depicts a practical example of how private and public identities relate to each other. Alice has two service profiles, one for work and one for home. The work profile consists of two public identities:

one for her email related services: *sip:alice@work.com*  
 and one for telephone related services: *tel:+47 12345678*.

The home profile has two corresponding public identities:

*sip:alice@home.com* and  
*tel:+47 91011121*.

It is technically possible for Alice to have several subscriptions. Subscription 2 with its minimum requirement of one Private Identity and one service profile is included for illustrative purposes. A more specific example will be revisited in the scenario in chapter 6 .

The public identity system in IMS is flexible. All Alice's public identities can be mapped to a single device, according to rules set by Alice, for example:

*"always route +47 12345678 to my smart-phone" AND  
"route alice@work.com to my smart-phone if WLAN is active"*

Alternatively, Alice's IMPUs can be shared among several devices:

*"route +47 12345678 to my cell-phone" AND "route +47 91011121 to my home-phone" AND "route alice@work.com to my laptop"*

For the third alternative that is characteristic of IMS, Alice can use a single IMPU for all her devices:

*"route alice@work.com to my cell-phone from 8:00 to 17:00" AND "route alice@work.com to my home-phone and after 17:00 and before 8:00" AND "route alice@work.com to my laptop"*

### 2.2.3 Public Service Identity

The Public Service Identity (PSI) is used to identify services, specific service features and user specified group lists (for example a Presence list). Note that although the resources identified are hosted on application servers, these identities may be created and modified by users.

Recall the User Equipment (UE) as pictured in figure 2.1. The UE communicates primarily with the P-CSCF using SIP but it can also communicate with a SIP AS over any HTTP based protocol such as XCAP. The UE communication over HTTP is particularly relevant to our discussion. It is here that an end user can configure information stored on an application server. Typical usage includes adding or updating resource lists on a Presence server. More details on use of Presence in IMS is given in chapter 6. In order to create a new resource list the UE must create a new PSI. The PSI is in the form of a URI and is used to identify the new service being created. Specific examples of this will follow in chapter 5.

### 2.2.4 Service Profile

Service profiles are stored in the Home Subscriber Server (HSS) database. Each profile is mapped to a user (IMPU) or to a service (PSI) and contains data necessary for routing requests originating from or terminating at that user/service. The service profile consists of a prioritized list of initial Filter Criteria (iFC), which can be stored in XML format. The iFC is a prioritized list of instructions for triggering services in a session. The concept of iFC is discussed in more detail in section 5.2.

## 2.3 Databases

Efficient data storage and management is in itself a large topic in telecom service provisioning. No less so in IMS. AS mentioned in 1.4 we will not discuss the topic of databases in depth here but we will present a basic overview of IMS databases involved in service provisioning.

### 2.3.1 Home Subscriber Server

The Home Subscriber Server (HSS) is the master IMS user database. It contains functionality for both IP Multimedia as well as for the GSM Home Location Register (HLR). The user-specific data stored here include user identities, service profiles, location data, and other information necessary for authentication and authorization. The HSS communicates with I/S-CSCF and SIP AS over the Diameter protocol.

For large capacity networks there may be more than one HSS. In this case, another database called a Subscription Location Function (SLF) is also present for resolving HSS addresses. The S-CSCF and I-CSCF will then query the SLF to find the correct HSS for a particular user's profile.

### 2.3.2 S-CSCF Media Policy Database

The S-CSCF Media Policy Database stores media policy information for each user profile is indicated by way of a single integer that is stored in the Core Network Authorization class. The HSS sends this integer (as a part of the service profile) to the S-CSCF over the Diameter protocol. For this reason, the S-CSCF has a static database that is used to map integer values to media profiles.

### 2.3.3 XML Document Management Server

An XML Document Management Server (XDMS) is a native XML database system standardised by the Open Mobile Alliance (OMA) [10]. It is used for user data management and group management. This database allows data to be shared across multiple applications and is particularly relevant in Presence implementations [11]. In IMS, the XDMS is typically deployed in the service layer as a part of an AS. As with all entities deployed within application servers in IMS, its use is not standardised by 3GPP.

## 2.4 Serving - Call Session Control Function

There are three Call Session Control Functions (CSCF) in IMS: the Interrogating CSCF (I-CSCF), the Proxy CSCF (P-CSCF) and the Serving CSCF (S-CSCF). The S-CSCF is central in IMS communications and specifically in service provisioning. It is therefore important to understand its functionality before analysing an orchestration framework.

The S-CSCF communicates with SIP application servers (SIP AS) using the SIP protocol and with HSS using the Diameter protocol. Different S-CSCFs in the same network can play any of the following SIP Server roles as specified by the IETF SIP specification [12]:

- Registrar: In this capacity the S-CSCF.
  - accepts registration requests that have been routed from UEs. It then passes this registration information on the HSS. The data the S-CSCF receives subsequently from the HSS is used to authenticate the UE by way of a challenge

response procedure. If the user is authenticated then the service profile associated with her current IMPU is downloaded from the HSS. This profile will later be used for service routing and media policy negotiations.

- notifies subscribers about registration changes (relevant for Presence services for example).
- controls sessions for registered users.
- Proxy Server: In this capacity the S-CSCF accepts requests, services requests internally (for example translates from MSISDN<sup>2</sup> to SIP URI) and/or forwards requests to another IMS entity (for example a P-CSCF or an I-CSCF)
- User Agent: In this capacity, the S-CSCF
  - Independently generates or terminates SIP transactions
  - Provides service-event related information to endpoints(also relevant o Presence)

### 2.4.1 Initial Filter Criteria

In the process of IMS service invocation, the selection and ordering of services is based on the filter criteria in the user's service profile. For example, when Alice first registers, the S-CSCF retrieves her User Profile from the HSS. Her user profile contains her three service profiles. Each service profile may in turn have its own prioritised list of initial filter criteria (iFC). Thereafter, each time Alice initiates or terminates a session, the S-CSCF will choose a profile based on information in the request headers and from this profile it will invoke the required services sequentially. A detailed scenario demonstrating how this is done is provided in chapter 6.

## 2.5 SIP Application Server

In the IMS domain, the SIP AS is the entity which hosts native services. One SIP AS may host several multimedia services, each identified with a PSI. Let us take an example of a SIP AS containing three services. These services could be: Presence [11], Instant Messaging and Conferencing. Each service can be invoked to add functionality to a SIP session.

The AS can access the user's service profile on the HSS and transparently store application data on the HSS. Reciprocally, the AS is alerted to changes made to application data stored on the HSS [13]. The data format for this profile is open. We can use this to our advantage by choosing XML as our data format. This gives us the possibility to employ XCAP and Web Services in our architecture.

As mentioned in section 2.2.3, UE and SIP AS can communicate directly via the XCAP interface. In such a scenario an XML-based user profile can be modified in order to customize user services or configure user terminals.

---

<sup>2</sup>Mobile Station International Subscriber Directory Number: a mobile telephone number

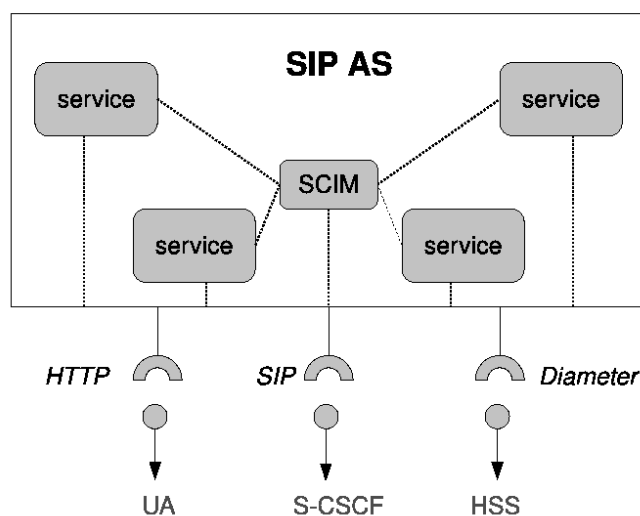


Figure 2.3: SIP Application Server

As with the S-CSCF, the SIP AS can play several SIP Server roles as specified by IETF SIP [12]:

- SIP Proxy
- SIP Registrar
- SIP Redirect Server (e.g.: redirecting a user to an external web service)
- SIP User Agent (acting on behalf of a client entity)
  - Originating (issues request, e.g.: setting up a multi-player online game)
  - Terminating (receives request, e.g.: accepting voice mail)
  - Back-to-Back (B2B)
- Third-Party Call Control (initiates a new dialogue)
- Routing (handles incoming and outgoing dialogue)

The redirect, UA and routing roles are particularly relevant to designing a new service orchestration architecture as we will illustrate in chapter 4.

## 2.6 Service Capability Interaction Manager

The IMS specification [8] refers to the Service Capability Interaction Manager (SCIM) as an application that provides interaction management. Here the term "application server" encompasses a module of several application servers. The SCIM is within this module as indicated in figure 2.3. Beyond this brief mention, the SCIM is not specified in IMS, leaving SCIM as a concept open to widely different interpretations and implementations.

Vendors that claim to offer SCIM functionality often define their SCIM product in the role of coordinating and mapping between native IMS services and services residing on non-native and legacy platforms. Specifically, the role of coordinating services between SIP AS,OSA and CAMEL application servers. However this interpretation of SCIM is a subject of heated debate and several other roles have been suggested/implemented for the SCIM, such as the list suggested by Palmeter in [14]. In chapter 4 we present a more thorough discussion of the SCIM and in chapter 5 we take a position on what role the SCIM should play as well as where it should be located.

In the next chapter we will discuss related enabling technologies in the context of telecom related service provisioning and composite services.

# Chapter 3

## Related Technologies

This section discusses architectures and frameworks relevant to further discussion on service orchestration in IMS. To give a visual background to this discussion, figure 3.1 illustrates one scenario in which these related technologies can interplay. The scenario is not IMS specific. Section 3.1 discusses Service Oriented Architecture(SOA). Section 3.2 discusses a subset of SOA called User Oriented Architecture(SOA). Section 3.3 discusses Event Driven Architecture, another possible subset of SOA. Section 3.4 discusses Web Services (WS), a technology that is typically used to implement SOA. Section 3.5 presents the concept of service orchestration, a technique that can be employed in the provisioning of composite WS. Section 3.6 introduces the concept of an Enterprise Service Bus (ESB), an element that could help provide an orchestration solution based on EDA. Section 3.7 introduces the general concept of Service Delivery Platforms (SDP) as a tool for service delivery and section 3.8 briefly discusses the Operational Support Systems and its role in telco service platforms. Finally, section 3.9 discusses Web 2.0 in the broader context of current trends in distributed services.

### 3.1 Service Oriented Architecture

In its working group note on WS architecture [1], the World Wide Web Consortium(W3C) refers to Service Oriented Architecture (SOA) as a form of distributed systems architecture characterised by its logical view, message orientation, description orientation, granularity, network orientation and platform neutrality. From their description we can then summarise a service in the SOA context as an entity that is:

- an abstract logical view of a process defined by what it does,
- formally defined in terms of platform-neutral message exchange between its provider and requester agents,
- described by machine processable meta data
- typically carrying out few operations but with large complex messages.

From this description, this architecture is clearly not appropriate for all distributed applications. Here is a list of qualities that might indicate that an application is appropriate for SOA, as interpreted from the above mentioned W3C note. Such an application:

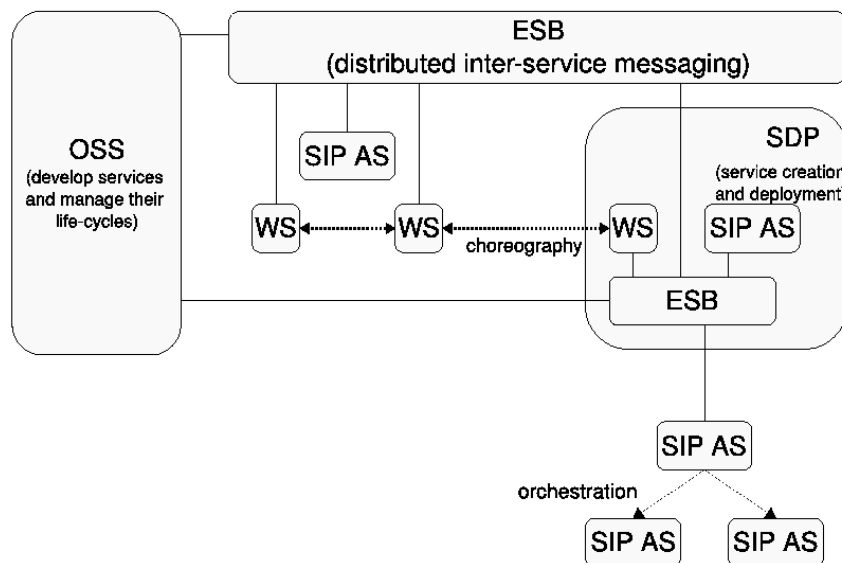


Figure 3.1: Relationship between the Enabling Technologies

- operates over the Internet (typically no QoS guarantees);
- does not have managed deployment mechanisms (group upgrades of requesters and providers);
- has system components that are to some extent vendor and platform neutral;
- already exists and needs to be exposed for use over a network (can wrap as a WS).

In chapter 4 we discuss some of the problems that arise from this list of guidelines.

## 3.2 User Oriented Architecture

User Oriented Architecture (UOA) is not yet widely mentioned in the IT world. UOA can be described as user-centric SOA. It can be argued that IMS is a UOA. due to the following facts:

- SIP is the central IMS protocol and SIP in itself has several user-centric features:
  - SIP is designed to put intelligence in the end-nodes and even with an IMS between users, this inherent quality is still there.
  - A SIP URI identifies a person rather than a device (unlike, for example telephone numbers)
  - The SIP event notification [15] extension is (PUBLISH,SUBSCRIBE,NOTIFY) coupled with the growing collection of standardized SIP event packages,

give users the power to create and publish their own service events as well as to monitor events. Presence is the most well known event-notification scenario but it is far from being the only one.

- IMS service chaining is steered by preferences set in the individual user's service profile.

Webalo <sup>1</sup> provides a UOA product called a User Proxy. Webalo claims that the User Proxy "achieves loose-coupling between users and systems". According to the information on their web site, the proxy is essentially a WS that other services utilize to interact with the user. Applications are dynamically customisable and appear to be tailor-made for the client to which they are downloaded.

### 3.3 Event Driven Architecture

Event Drive Architecture (EDA) are loosely coupled and distributed by nature and are therefore best suited to asynchronous systems. There is one style of EDA can be described as a style of SOA in the sense that either an event occurrence can trigger one or more services in a SOA environment, or, a service in a SOA can trigger the occurrence of one or more events [16]. However EDA covers a much broader scope of architectures than just event driven SOA.

In this context, we define an event as a significant change of state [17]. In order for the event model to work well, the normal boundaries of state behaviour must be well defined so that significant changes of state can be identified (expectation → deviation → response). Boundaries should be explicit, such as upper and lower time thresholds or a certain percentage of CPU time being idle. The event chain consists of an event source, an event channel, an event processing unit and an even driven activity.

SIP is a good candidate for implementation on EDA platforms because it is itself an event-oriented protocol with clear states. For example, SIP defines four different transactions: INVITE Client transactions, INVITE Server transactions, Non-INVITE Client transactions and Non-INVITE Server transactions. Further, depending on the combination of method type (INVITE or non-INVITE) and peer role (client or server), each transaction may be in or transition to one of six explicit states at any given time, namely: calling, trying, proceeding, completed, confirmed or terminated [12]. Figure 3.2 depicts a simplified state machine for the INVITE client transaction. State transitions are labeled with the events that trigger them. Triggers may be requests received (INVITE in this case), responses received (three digit status codes such as 200 which means "OK" or 180 which means "ringing"), an error signal or a timer signal. The action taken with each trigger is not shown in this diagram. In this case the event source is the SIP user agent or SIP stateful proxy; the event channel is the transaction itself and the event driven activity is the execution of the SIP method. For the sake of completeness we can also mention that SIP also defines explicit states for dialogs.

Another example of a framework that implements an EDA is a Service Level Execution Environment (SLEE). We will discuss SLEE further in chapter 3.7.3.

---

<sup>1</sup><http://www.webalo.com/technology.html>

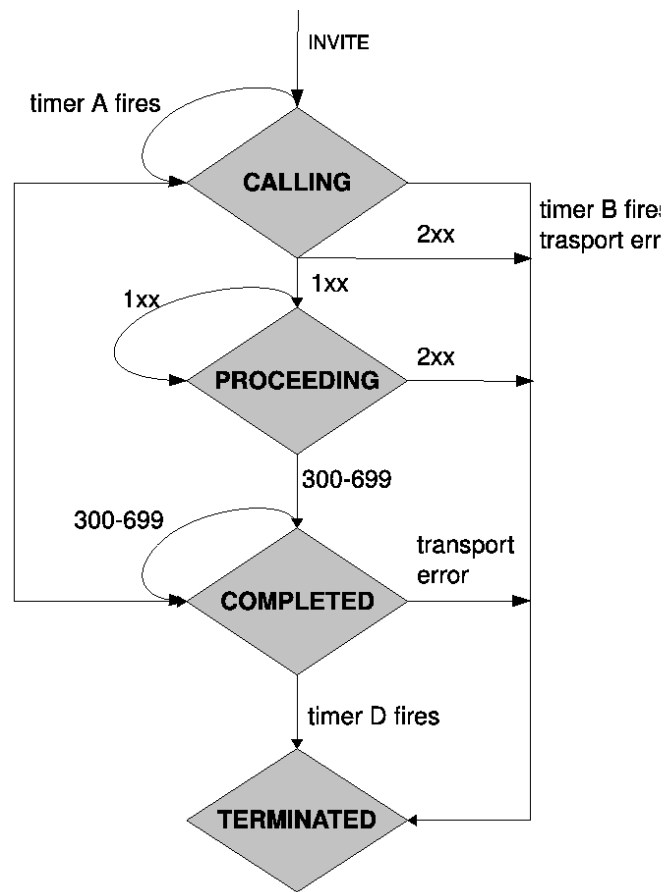


Figure 3.2: SIP INVITE Client Transaction - state machine

### 3.4 Web Services

SOA can be implemented without the use of WS and WS can be used without implementing SOA; yet WS is a technology typically used to implement SOA. There is no one WS specification, there are several, from a variation of commercial and standardization organisations. As there is no one specification, neither is there one authoritative definition. We will use W3C's technical report on WS architecture [1] as a guideline. Supporting specifications from W3C and OASIS are referenced as needed. The Web Services Interoperability Organization (WS-I) provides what they call profiles which are a set of specifications aimed at best practices for WS interoperability. Figure 3.3 gives an overview of typical Web Services interactions.

Based on the various descriptions and implementations of WS here is a summary of protocol-agnostic requirements:

- A web service must be capable of interacting and coordinating with other services as well as with service consumers
- It must be discoverable and consumable over a network. This implies that:

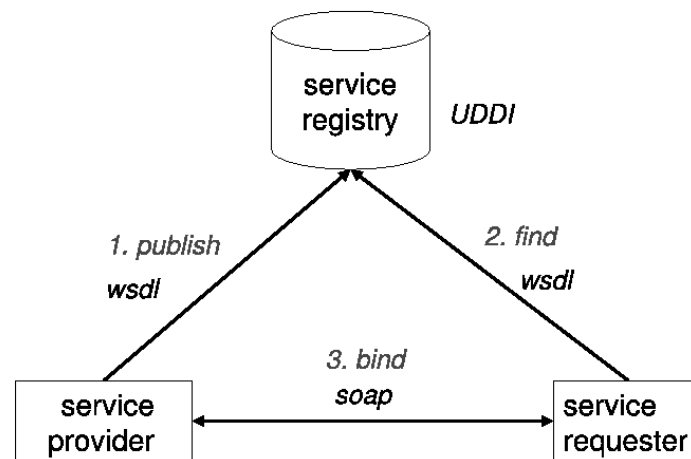


Figure 3.3: Standard Web Services Interactions - Overview

- It must expose an interface to potential service consumers.
- The exposed interface must describe the service in terms of its capabilities and information needed to communicate with it.
- The service description must be machine-processable.

These requirements are summarised in one of the more concise WS definitions presented by Haas<sup>2</sup>:

*A Web service is a software system identified by a URI, whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols.*

According to common practice in WS development we can say that WS is partially defined by the protocols and interfaces typically used in implementing these requirements. A presentation of said protocols follow in section 3.4.1 through 3.4.4 . Figure 3.4 gives an overview of a typical WS protocol stack. Note that though SOAP appears to be the de facto messaging protocol in WS, it is not a requirement. We mention several alternatives to SOAP in section 3.4.4 and discuss one of the alternatives in detail in section 3.4.6. In traditional world wide web scenarios web applications are consumed by end nodes using HTML as the message protocol. In WS, services are consumed by web applications, using XML as the transport protocol.

### 3.4.1 Extensible Markup Language (XML)

XML is an open and extensible data object format that is the basis of several of the WS protocols. It is standardised by W3C<sup>3</sup>. One of XML's main attractions is the ability it

<sup>2</sup><http://www.w3.org/2003/Talks/0521-hh-wsa/slide8-0.html>

<sup>3</sup><http://www.w3.org/TR/REC-xml/>

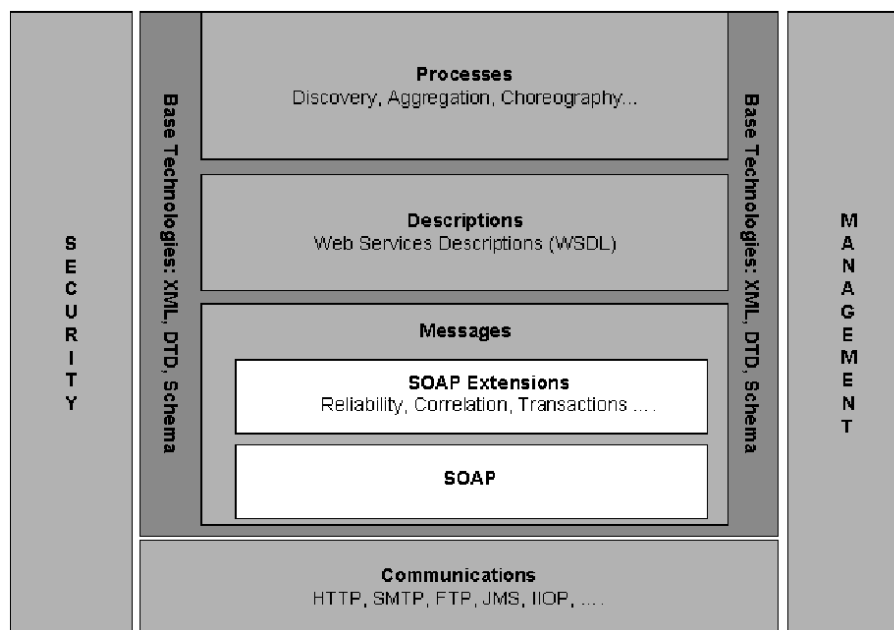


Figure 3.4: Typical Web Services Protocol Stack [1]

provides for using the same data structure across heterogeneous systems. This makes it particularly attractive for use across the Internet. In the WS context it is used to serialise data. However, the specifications and standards that describe XML in its entirety are vast, complex and ever growing. Its important features as utilised in WS are:

- XML Namespaces (xmlns): URI References that globally and uniquely qualify element and attribute names that might otherwise be ambiguous. For example `xmlns:env="http://www.w3.org/2003/05/soap-envelope"` defines W3C's XML namespace for soap envelopes
- XML Infoset: a specification that provides an abstract description of different parts of a well-formed XML document for reference in other XML specifications. Where a *well-formed* XML document is defined as one that has correct XML syntax.
- XML Schema Definition (XSD): A set of conformance rules to be applied for an XML document to be considered valid, where a *valid* XML document is defined as a well-formed XML document which also conforms to an XML schema. For the sake of clarity, note that "XML Schema" is the name of W3C's XML schema and XSD is an instance of "XML Schema".

In our problem formulation in chapter 4 we will discuss some of the limitations of XML for applications in telecom environments.

### 3.4.2 Web Services Description Language (WSDL)

WSDL is an XML language that is used to model and describe web services [18]. WSDL describes a service abstractly in terms of its message exchange patterns as well as concretely by specifying the protocol (e.g., SOAP) and the address (e.g., a URI) of the service port (in other words by specifying its binding information). It is important to note

that WSDL 2.0 accepts binding to all HTTP request methods. WSDL 1.1 only accepted binding to the POST and GET methods. The consequence of this change is better support for RESTful Web Services. REST is further discussed in section 3.4.6. As WSDL 1.1 is not recommended by W3C we will curtail our discussion to WSDL 2.0.

In practical terms, WSDL allows a service to advertise its capabilities to potential clients. The client uses the information presented in the exposed WS interface to bind to, consume and interact with the service.

### 3.4.3 Universal Description Discovery and Integration (UDDI)

It is fully possible to have a functional WS architecture without the implementation of a WS discovery service. However, for anything other than small private networks with few services it is not practical. Without a discovery service, clients must have extensive knowledge of all available services in order to find, bind to and consume them. This would create an unacceptable overhead for a telecom system that wants access to a wide variety of services over the Internet so it is dismissed as infeasible.

The WS architecture does not specify how a WS discovery service obtains or handles service descriptions. However, UDDI is one popular option. It is standardised by OASIS as "*a 'meta service' for locating web services by enabling robust queries against rich metadata*" [19]. The current specification is UDDI 3.0 and it is designed to provide for building "*flexible, interoperable XML Web services registries*". XML schema are used to describe the UDDI data structures. The discovery process can be done at design-time or at run-time and must and can be autonomous or manual. These are important differences to consider in WS system design. We will discuss this further in the context of IMS orchestration requirements in chapter 4.

### 3.4.4 SOAP

SOAP is a protocol designed to facilitate the exchange of structured messages in a distributed network. W3C's SOAP 1.2 specification defines an XML-based messaging framework as well as three optional components, namely: (1) encoding rules for expressing application-defined data types, (2) SOAP's remote procedure calls (RPC) conventions, and (3) SOAP's HTTP/1.1 conventions.

SOAP messages can be sent within or on top of a variety of network protocols such as HTTP, SMTP, FTP, RMI, or even proprietary protocols. HTTP is the most widely used and the only one we will refer to in our further discussion. The specification provides general rules for binding SOAP to different protocols. While other distributed messaging technologies such as the Distributed Component Object Model (DCOM) and the General Inter-Orb Protocol (IOP) are generally filtered by firewalls, SOAP tunnels through firewalls without problems. This presents a significant security risk for IMS Service Providers who want to open up their platforms to WS orchestration. In addition the basic SOAP specification provides no security mechanisms. In other words, access control, confidentiality, integrity and non-repudiation are not addressed. Security in SOAP implementations can be added optionally as WS extensions.

A SOAP message is defined as an XML infoset which consists of a mandatory *envelope* item which is the message container and within the envelope, a mandatory *body*

item which the actual message being transmitted and an optional *header* item which contains application specific information.

### 3.4.5 Web Services Invocation Framework

The Web Services Invocation Framework is a Java framework for invoking WSDL-described services regardless of the protocol being used for message exchange. This means with use of WSIF, one can invoke web services without the use of SOAP.

### 3.4.6 Representation State Transfer (REST)

Agents identify objects in the system, called resources, with Uniform Resource Identifiers (URI). Agents represent, describe, and communicate resource state via representations of the resource in a variety of widely-understood data formats (e.g. XML, HTML, CSS, JPEG, PNG). Agents exchange representations via protocols that use URIs to identify and directly or indirectly address the agents and resources.

REST is an architectural style for reliable Web applications first described by Fielding in [20]. According to the thesis, Fielding's intention was to present a model of how the modern web should work. It provides principles which outline how resources on the web are defined and addressed. Further, application of REST architecture on the Web is meant to improve interaction scalability, reduction of interaction latency, generality of interfaces, independent component deployment, security enforcement and legacy support. REST web agents provide uniform interface semantics (create, retrieve, update, delete) rather than arbitrary or application-specific interfaces, and manipulate resources by exchanging representations. The messages do not depend on the stored state on the server, so the interactions are said to be stateless. W3C identifies REST-compliant Web services as those whose primary purpose is to manipulate XML representations of Web resources using a uniform set of "stateless" operations as opposed to arbitrary Web services, that are identified as which may expose an arbitrary set of operations. SOAP 1.2 can be used in a REST compliant manner, however SOAP adds a layer to HTTP whereas REST in itself does not. This leads us to following criticism of SOAP. W3C has been criticised for promoting SOAP as a WS standard and for not trying to promote REST.

A frequent criticism of the SOAP standard is the verbosity of its messages. As a result RESTful is often recommended by SOAP critics as a way to minimize the messaging overhead. An example of the difference is shown in table 3.1: the same WS request for a Flickr service called flickr.test.echo<sup>4</sup>. is first implemented in a RESTful fashion and then the same request is implemented using SOAP. One interesting observation is that while REST architecture stresses uniform interfaces SOA stresses varied described interfaces.

There are several indications from Marketplace leaders that suggest that REST is gaining ground over SOAP in large WS implementations. Here is a list of some of those indications:

---

<sup>4</sup><http://www.flickr.com/services/api/>

<b>Request service "flickr.test.echo" with SOAP:</b> <pre> &lt;s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope" xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance" xmlns:xsd="http://www.w3.org/1999/XMLSchema"  &lt;s:Body&gt; &lt;x:FlickrRequest xmlns:x="urn:flickr"&gt; &lt;method&gt;flickr.test.echo&lt;/method&gt; &lt;name&gt;value&lt;/name&gt; &lt;/x:FlickrRequest&gt; &lt;/s:Body&gt; &lt;/s:Envelope&gt; </pre>
<b>Request "flickr.test.echo" with REST:</b> <pre> http://api.flickr.com/services/rest/?method=flickr.test.echo&amp;name=value </pre>

Table 3.1: SOAP vs. REST

- In 2003, it was reported that 85% of calls to the amazon.com Web Services were done over their REST interface (they provide a SOAP interface as well)<sup>5</sup>.
- In 2006 Google depreciated their SOAP Search API in favour of an Ajax Search API<sup>6</sup>.
- The highly popular Web 2.0 services like those mentioned in section 3.9 all use RESTful interfaces. A few of them, like www.flickr.com, provide a SOAP interface as an option but the majority of them don't.

It is worth remembering that SOAP-based WS can be RESTful as well as that WS technologies can be useful in the context of non-SOAP-based REST services. Examples of technologies on the web that are RESTful include Really Simple Syndication (RSS) and Asynchronous JavaScript and XML (AJAX), both technologies that are heavily used in Web 2.0 mashups. For further discussions on Web 2.0 and mashups see section 3.9.

### 3.4.7 OSA Parlay X

Parlay X is set of telecom Web Services offered by the Open Services Architecture (OSA) group. Recall from section 1.4 that the OSA SCS [21] is one of the two optional application servers supported in IMS. Through this server, IMS gains limited support for Web Services. So, though out of scope for our implementation it is discussed here due to its relevance to WS.

<sup>5</sup><http://www.oreillynet.com/pub/wlg/3005>

<sup>6</sup><http://radar.oreilly.com/2006/12/google-deprecates-their-soap-s.html>

The OSA SCS in the IMS service layer provides limited WS capability to the IMS domain. It is attached to an external OSA WS gateway via the OSA API. The WS gateway in turn connects to an OSA AS over standardised Parlay X interfaces as well as publish to a WS Registry using standard WS interfaces. It can be argued that an advantage of using OSA to access external WS is that OSA SCS is in itself a secure gateway and therefore security features (authorisation and authentication) are built into its architecture. However we are attempting to find a native service orchestration solution for IMS which has additional capabilities to include a wide variety of (external and internal) web services in its service compositions.

There are several limitations to using the OSA approach to implement Web Services in IMS. Firstly, according to the current IMS specification [8], the OSA SCS is the only OSA element specified within the IMS domain. Since the OSA SCS is only a gateway, this implies that none of the OSA/Parlay services can be hosted inside the IMS domain. In order to use Parlay to implement native SIP messages, all SIP proxies involved must be call-stateful and all SIP requests must be mapped to Parlay requests [22]. Cursory consideration implies that mandatory mapping of all SIP requests within a native SIP environment is redundant and inefficient. Refer to section 4.5 for a discussion of the negative effect of stateful proxying on performance. So the OSA application cannot participate in SIP signalling flow. Instead, SIP requests are mapped to OSA API invocations via the OSA SCS. OSA services are typical telecom services such as 3rd party call control and SMS. Given that an IMS would be under the control of a telecom operator, it is counter-intuitive to adopt an architectural feature that does not allow typical telecom services to be hosted in the IMS. From a business standpoint, this is clearly not an attractive option for an IMS provider/operator and we argue that it serves as a deterrent against wide adoption of the OSA CSC in IMS. Secondly, the general OSA API is not based on widely used WS standards. It is based on CORBA and it is deemed to be complex. Again, this serves as a deterrent against wide adoption. Thirdly, Parlay X Web Services is limited to a finite set of telecom services, which for version 2.1 (equivalent to 3GPP TS 29.199-1 through 14) are namely:

- Third Party Call,
- Call Notification
- Short Messaging (SMS),
- Multimedia Messaging,
- Payment,
- Account Management,
- Terminal Status,
- Terminal Location,
- Call Handling,
- Audio Call,

- Multimedia Conference
- Address List Management and
- Presence

In summary, we conclude that the OSA SCS component in the IMS service layer is useful only as a gateway to existing telecom services implemented in OSA/Parlay. It is not suitable for native IMS service orchestration nor is it suitable for interfacing with generic Web Services. Whether there is a need for 3rd-party, non-SIP, telecom services in composite IMS services can also be disputed.

### 3.4.8 Web Services Security

For a business to deploy applications to a WS environment imply that they must expose interfaces to some vulnerable internal resources. Both stored data and data being transferred over communication channels would become potential targets.

As mentioned in section 1.1 the basic WS architecture does not include mandatory security. The WS-I standard recommends HTTPS security which thought possibly sufficient for some IT services is not sufficient for the IMS platform. We have also discussed SOAP's ability to sneak through firewalls unnoticed. This is a point that would have to be carefully addressed in any native IMS WS implementation. OASIS (<http://www.oasis-open.org>) is a pioneer in developing important extensions to the basic WS specifications. The most important of these is arguably the WS-Security specification. At time of writing the latest version was WS-Security 1.1 from November 2006. It focuses on enabling secure SOAP transactions via the XML-based, Security Assertion Markup Language(SAML), Rights Expression Language (REL) and kerberos.

The WS-Reliability specification is also from OASIS and provides standards for reliable WS messaging. In addition, the WS-Transaction specification is provided by BEA Systems. While security, reliability and transactionally sound architecture is extremely important for IMS it is also a topic in itself that requires its own treatment. So for this discussion we will suffice to say that any WS implementation adopted in the IMS domain should have mandatory WS security extensions built into its specifications. The ones mentioned here from OASIS and BEA are a good starting point.

## 3.5 Service Orchestration

Recall from section 1.1 that the term "service orchestration" typically refers to Web Services orchestration. In this context, WS orchestration automates interactions between loosely-coupled business processes that together represent a composite web service.

### 3.5.1 Choreography vs Orchestration

It is useful to note the differences between service orchestration and service choreography. The difference may seem subtle because they both deal service interactions, but the functional differences are important.

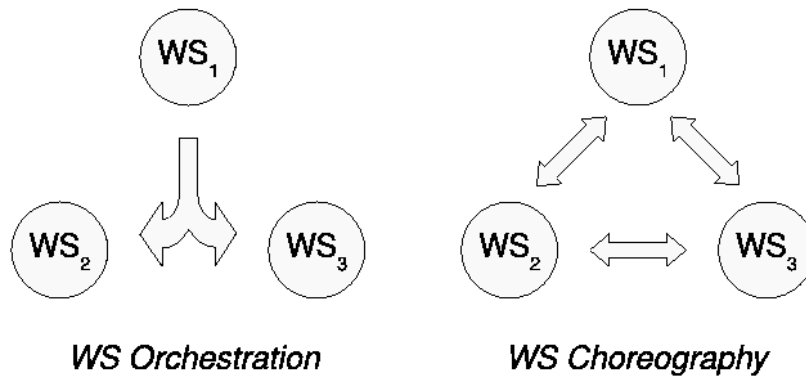


Figure 3.5: Choreography vs Orchestration

According to [23] choreography refers to collaboration between processes where all parties have a similar level of influence on the composite product while orchestration refers to one steering process which gives instructions to other processes that are necessary for its composite product. There are clear parallels to the difference between orchestration and choreography in the music world. Figure 3.5 illustrates the differences.

Recall from section 1.3 that we have chosen to approach our discussion from the perspective of the operator. From this perspective - that is, that of a telco provider and controller of an IMS domain - it is the orchestration of services that is of immediate importance. However, one must note that orchestration and choreography are not mutually exclusive. They are, for example, both necessary elements in a complete web services scenario. An example of orchestration in WS is modelling of participant behaviour in a single workflow as done in executable processes in the Business Process Execution Language (BPEL). An example of choreography in WS is the public message exchange between independent parties in a BPEL abstract process.

### 3.5.2 Limitations

The challenge of applying WS orchestration in our context is that this type of orchestration is that Web Services are inherently non-real time.

## 3.6 Enterprise Service Bus

, The term Enterprise Service Bus (ESB) refers to a logical architecture that is used to communicate with distributed, abstracted services. An ESB is responsible for passing messages between the services that are attached to it. It is usually used in an enterprise environment between services that represent business processes.

An ESB can be classified as an EDA. In fact, A SLEE can be considered to be a

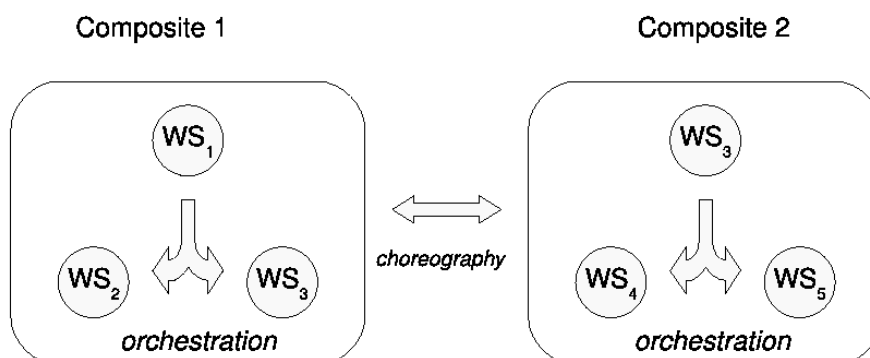


Figure 3.6: Choreography and Orchestration

type of task-oriented ESB<sup>7</sup>. We take a closer look at a specific SLEE implementation in chapter 6.

## 3.7 Service Delivery Platforms

At time of writing there was no standard definition for SDP. However the Telemanagement Forum (TMF) is in the process of standardising such a definition. One of the most recent efforts to this end was the TMF Service Delivery Platforms Summit in June/July 2008. One of the aims of TMF appears to be to create a strong association between SDP and SOA. One definitions given of SDP in summit literature was a "framework for exposing, managing and controlling Service Provider network assets". The motivation for utilizing Service Delivery Platforms (SDP) is to achieve better integration of telecom infrastructure for rapid development of converged services. Service creation, orchestration and execution are key steps. SOA by way of web services is used sparingly in SDP, specifically for business systems. SDPs are offered not only by telecom companies but also by traditional Internet companies and system integrators. A Service Level Execution Environment (SLEE) can be considered to be a type of SDP.

### 3.7.1 Telephony Application Programming Interface (TAPI)

TAPI is a computer telephony integration API developed by Microsoft and Intel. It is integrated with Microsoft operating systems (since Windows 95) and allows the use of telephone services on individual computers and computer networks. The Java telephony API (JTAPI) provides a cross platform solution for telephony call control.

### 3.7.2 SIP Common Gateway Interface(SIP CGI)

The original CGI specification was developed by W3C to facilitate service creation and deployment in a Web Services environment. Concretely it serves as an interface to

<sup>7</sup><http://gradecak.blogspot.com/2007/09/jain-slee-or-esbjbisca.html>

HTTP Servers. Given the parallels between HTTP and SIP, SIP CGI was standardised by IETF (RFC 3250) for the creation and deployment of services in a SIP environment. Interfacing in this case with SIP Servers.

### 3.7.3 Service Level Execution Environment

JAIN SLEE or JSLEE is the Java implementation of a SLEE. It has an EDA which by definition indicates its support for applications that require low latency, high throughput signalling [24]. In fact, JSLEE has put an upper latency limit that must be met by JSLEE certified application servers. JSLEE 1.0 (JSR 22) was finalised already in 2004 while the final release of JSLEE 1.1 (JSR 240) came July 15, 2008.

JSLEE's greatest advantage is perhaps its flexibility. It is modular. It supports many network protocols. It upholds ACID (Atomicity, Consistency, Isolation, Durability) properties and it is event driven. These properties make it a strong candidate for development of composite telecom services. However, its complexity may be a deterrent for many developers. Developers who are already at home in complex J2EE environments or/and who have experience with Enterprise Java Beans (EJB) may be better at adopting JSLEE.

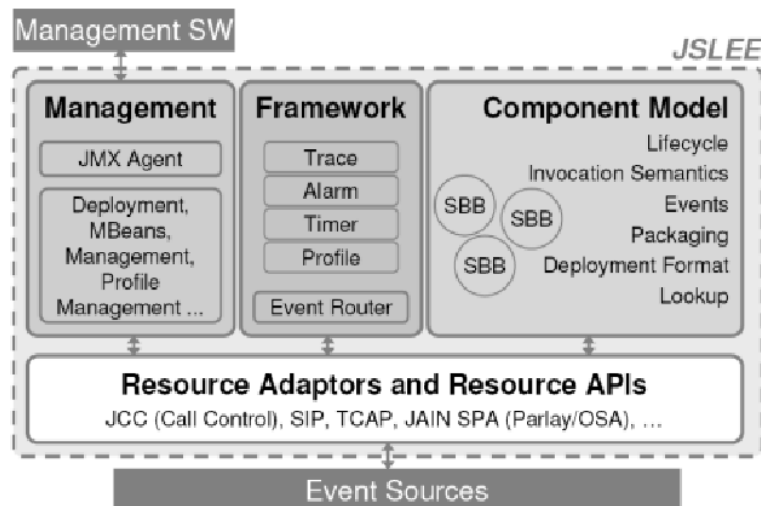


Figure 3.7: Overview of JSLEE Architecture [2]

### 3.7.4 SIP Servlets

As of version 1.1, SIP Servlet has adopted a central application invocation logic that is well suited for IMS application chaining. This new logic is implemented by way of a new component in the Servlet container called the Application Router (AR).

For a given framework, all SIP Servlet applications have only one main Servlet to relate to. The AR is a central component that interacts with the service container through a Java interface. On start-up, the container sends an initial query to the AR. The AR then uses a selection of relevant information it has access to (for example SIP headers, user profiles or service profile) to choose and return the name of the first

application to be invoked. It also passes state information to the container at this time. The container can then use this information to make further requests as necessary. Note that this functionality matches the IMS specification for service provisioning through implementation of initial Filter Criteria (iFC). Recall that concept of iFC was introduced in section 2.2.4. An iFC model is presented in chapter chap:model.

The greatest advantage of the SIP Servlets is perhaps its simplicity especially when contrasting it with JSLEE. It is light-weight and integrated with the SIP stack and its model resembles that of HTTP Servlets, which SOA developers are already familiar with.

### 3.7.5 SDPs and IMS Requirements

In [5], Khlifi and Gregoire present a useful overview on available implementation technologies for IMS application servers and how they fit into IMS service requirements. The discussion presented provides a suitable framework for choosing a technology for the SSD platform. A list of six requirements are used to measure SIP Programming techniques as exemplified by SIP Common Gateway Interface (CGI) and SIP Servlets, versus Application Programming Interfaces (API) as exemplified by OSA Parlay and Telephony API (TAPI), and service logic execution environment (SLEE) as exemplified by JSLEE. We omit TAPI from our discussion as being clearly too limited. For the remaining contestants, all options scored equally well on *range of service support* and on *universal service access*. On *ease of service customization* all options received an equally ambiguous score of "possible". For the criteria on which they differed, the table 3.2 summarizes the differences.

Feature	SIP CGI	SIP Servlet	Parlay API	JSLEE
rapid service creation	no	medium	yes	yes
multilayer support	no	limited	medium	high

Table 3.2: Overview of SDP Features (adapted from [5])

## 3.8 Operational Support System

NGOSS is the Next Generation Operational Support System (OSS) being developed by TMF. According to the specification, the model is loosely coupled, distributed and component based. NGOSS's Shared Information/Data model (SID) is an object model defined in UML. It is of particular interest to our discussion because although data must be shared between orchestrated applications, it is a challenging task. NGOSS's central approach is to set a common standard for shared data. However, though promising, SID is to date more suited to business models than to network

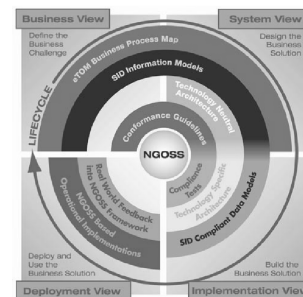


Figure 3.8: NGOSS

models. It is still a work in progress as the rest of NGOSS is. Figure x gives an overview of the NGOSS framework. Two specific motivations for NGOSS are particularly relevant to this discussion. They both imply better cost-effectiveness.

1. For service providers: A more modular OSS that can quickly develop and manage increasingly complex systems
2. For system integrators: More reusable elements and quicker adaptation and deployment of new elements

## 3.9 Web 2.0 Principles

The term *Web 2.0* can be considered a so-called buzzword used to describe recent trends on the world wide web. It is not an architecture, nor is it a specification, nor is it even an official term of any sort. Never the less, there have been distinct changes in the nature in which services are provided and consumed over the Internet and these changes have an impact on traditional service providers. It is therefore deemed relevant to a discussion on WS orchestration. Since an official definition is unavailable, a representative explanation of the concept seems more fitting:

Tim O'Reilly made an attempt at a concise definition of Web 2.0 in [25]. Though the resulting definition is not so concise, the principles mentioned seem to capture the Web 2.0 concept well. The terms "*harnessing collective intelligence*" and "*Don't fight the Internet.*"<sup>8</sup> are two descriptive phrases from Web 2.0 evangelism that are used by O'Reilly as well as his own quote from a 2005 description of web 2.0: "Data is the Next Intel Inside". about (A search for "harnessing collective intelligence" and "Web 2.0" gave 25,300 hits on a Google search). Further, in the following list, paraphrased from [25], O'Reilly offers these abstract principles for building services that are compatible with the Web 2.0 concept:

*"Don't treat software as an artifact, but as a process of engagement with your users.*

*Open your data and services for re-use by others, and re-use the data and services of others whenever possible.*

*Don't think of applications that reside on either client or server, but build applications that reside in the space between devices.*

*Remember that in a network environment, open APIs and standard protocols win, but this doesn't mean that the idea of competitive advantage goes away.*

*Chief among the future sources of competitive advantage will be data, whether through increasing returns from user-generated data, through owning a namespace, or through proprietary file formats."*

### 3.9.1 General Characteristics

Some important features of Web 2.0 services are summarised in the following list:

- **Social-\***

Web 2.0 services are inherently social in nature and are named as such. Some

---

<sup>8</sup>attributed to Eric Schmidt, Google CEO

social-\* concepts include social-networking (e.g., Facebook<sup>9</sup>, linkedin.com), social-bookmarking (e.g., del.icio.us<sup>10</sup>) and social-wallpapering (e.g., socwall<sup>11</sup>).

- **Dynamic in Nature**

Web 2.0 services are never static. Here is a list of some its dynamic service elements that illustrate this key aspect:

- **Web Feeds**

This is a method of publishing frequently updated content to users that subscribe to such alerts. Content providers syndicate content and content consumers collect an aggregation of updates from different sources into one application (called a newsreader). Newsreaders may be web-based or desktop. Really Simple Syndication (RSS) is one popular feed format for syndicated news. Facebook provides a proprietary news feed of what friends were last "seen" doing on Facebook (e.g., "Bob updated his profile picture", "Alice added Bob as a friend")

- **Tags**

Tagging is the use of keywords to describe web content such as pictures, parts of a picture (e.g. a specific person in a photo) and bookmarked web pages. Tags are usually subjective and a single resource usually has several tags. Users of social networking sites like *Facebook*, social bookmarking sites like *del.icio.us* and picture sharing sites like *Flickr* use tagging extensively. In fact, tagging is an integral part of *del.icio.us*' and *Flickr*'s concept.

- **Collaboration**

This is a reference to services that allow many individuals to collaborate on the development of a composite product. Example applications include wikis which are collectively edited web pages, the most well known of which is [www.wikipedia.org](http://www.wikipedia.org), a heavily used wiki encyclopedia.

- **User Generated Content and Services**

The fact that content and services are now actively generated by Internet users is perhaps the most revolutionary part of the Web 2.0 concept. Here are some concrete examples:

- The afore-mentioned **tags** are user generated.
- The afore-mentioned **wikis** are user generated.
- **Podcasts** (syndicated media files available over the Internet)
- Web logs (**blogs**) were one of the first Web 2.0 trends. Special blog formats include audio-blogs (usually as podcasts), video-blogs, photo-blogs and micro-blogs (e.g., [twitter.com](http://twitter.com)), micro-blogs are for sending very short updates such as "I'm hungry" or "just woke up". There is usually a very low restriction on maximum number of words/characters. It is therefore suitable for a mobile-blog (verb: **moblogging**) where updates or "tweets" as they

---

<sup>9</sup>[www.facebook.com](http://www.facebook.com)

<sup>10</sup>[www.delicious.com](http://www.delicious.com)

<sup>11</sup>[www.socwall.com](http://www.socwall.com)

are called in Twitter can be easily published via SMS. There is a plethora of freely available blog platforms on the web which make it easy for even the less technically inclined to create and maintain blogs. Popular alternatives include [blogger.com](http://blogger.com) and [wordpress.com](http://wordpress.com).

- The chance for users to develop and deploy their own loosely coupled services is becoming increasingly popular. Facebook, Myspace, Google and Apple are some of the pioneers in this field. Generally speaking, the platform in question offers an API that customers use to develop compatible applications. Licencing and ownership of the finished product varies.

### 3.9.2 Mashups

A Mashup is a hybrid web application that combines application data from several sources into one blended offering. The main advantage of Mashups is that the unique hybrid offers a service that could not have been offered singly by any the constituent services. As long as the appropriate interfaces are available from the potential services and APIs are available for interacting with them then only imagination and time restrict what kind of mashups are possible. The increasing popularity of mashups has even give rise to mashup editors such as Microsoft's Popfly <sup>12</sup>.

A so-called telecom mashup combines services from different providers into one integrated user experience. Service delivery platforms (SDP) are used to develop and deploy telecom mashups. Telecom mashups are deemed to be complex and present a challenge, especially in light of security, privacy and low-latency requirements of typical telecom services. In addition, telecom services are built on business models that rely on billing for the use of services, whether individually or as subscription package. Mashups evolved on the Web which is a much more open environment, its projection onto the telecom domain provides a billing challenge but it is an issue that must be addressed soon as telecom trends move towards all-over-IP.

---

<sup>12</sup>[www.popfly.com](http://www.popfly.com)

# Chapter 4

## Problem Formulation

In chapters 1 through 3 we presented some background information for the problem we would like to research and gave an overview of IMS as well as of supporting technologies and enablers for implementing a service orchestration architecture. In this chapter we will use the information we have gathered to discuss open issues, more narrowly formulate the specific problem and form a hypothesis for how it can be improved.

We start in section 4.1 by listing the problems faced in distributed systems in general. Section 4.2 presents some general requirements for IMS service provisioning. Section 4.3 discusses potential implementations of the SCIM. Section 4.3 discusses dynamic service brokering. Section 4.4 discusses how and where in the IMS domain such orchestration should be carried out. Section 4.5 discusses the performance in such an architecture. Finally section 4.6 summarises all the issues discussed into a concrete problem and section 4.7 presents a hypothesis on how to approach it.

### 4.1 General Challenges in Distributed Systems

Some of basic challenges to be considered in the design of an IMS service orchestration platform are inherently challenges of distributed systems. It is therefore appropriate to begin our discussion with a recap of challenges that are intrinsic to any distributed system [1]:

- The underlying transport medium may be unreliable or too slow.
- There is no shared memory between the caller and object.
- Partial failure scenarios: If only some service modules fail, or perform sub-optimally can we then say that the composite application is still available? How do we measure partial availability in a distributed application?
- Handling concurrent access to remote resources is complex.

## 4.2 IMS Requirements for Service Orchestration

To achieve dynamic service orchestration that is compatible with the IMS architecture while at the same time allowing the incorporation of non-IMS third party services, one must conform to a minimum set of requirements. The ability to provide rich multimedia services is a key part of the IMS principle as its name suggests. Today, that implies the ability to provide so-called quadruple play services: voice, video and data over high-speed broadband and mobile are offered through the same service provider. The typical scenario now is triple-play (no mobility support) where for example a cable company offers cable TV, cable internet access and a fixed telephone service in a bundle. However, the trend is towards quadruple-play and many providers have already adopted or are in the process of adopting this new business model. So what are the basic requirements for quadruple play over IMS? Some of these suggestions are based on the discussion in [26]

- The services should remain user centric: This remains true to the spirit of the SIP protocol and the use of service profiles in IMS subscriptions. User-centric qualities would include, for example, ability to control services directly (such as Digital Video Recorder functions while watching television) and as much as possible allowing the user to steer the level of automatic and adaptable behaviour through service/user profile settings.
- Service discovery mechanism: The system should use standard mechanisms for the discovery of IMS compatible services
- Content provisioning interface: There should be a well defined interface for third party content provision covering both push and pull methods. This should cover not only commercial content like IP TV but also content from other users for example from social networking sites.
- Quality of Service (QoS) support: The service provisioning system should be adaptive and should especially take into consideration the fact that service modules are from different providers from whom customers have different expectations of quality a availability [27].
- Service continuity while roaming: Given that IMS supports network convergence, ideal services should leverage this ability by allowing a continuous user experience as users change network types and domains.
- ability for 3rd party service and content providers to access data needed for charging
- Security: SAML should be mandatory for XML communication across security domains

## 4.3 The SCIM as a Service Broker

To address this limitation, a flexible and powerful service broker functionality could be introduced. To establish which features such a service broker would need, a good start-

ing point for this process is to review the potential capabilities of the Service Capability Interaction Manager (SCIM).

Recall our introduction to the SCIM in section 2.6 where we established that the SCIM is scarcely mentioned in the IMS specifications and that the role of the SCIM in vendor implementations vary widely. The SCIM role of interest for our discussion is that of the SCIM as a service broker. There a 3GPP technical report that discusses architecture impacts of service brokering [28] on IMS. Although not advanced to the point of specification, this report studies possible enhancements to the current (limited) service interaction management architecture in IMS. Although it is only preliminary, this study presents a natural starting point for our service brokering analysis. Sections 4.3.3 discusses service brokering options extrapolated from preliminary points presented in this report. As the service-broker SCIM is the only type of SCIM considered for the remainder of this thesis it will hereon in be referred to as "service broker" or SB for short.

### 4.3.1 Dynamic Service Brokering

To understand the limitations we face in IMS service provisioning let us revisit the S-CSCF which was introduced in section 2.4. The S-CSCF's main responsibility is session control. It maintains session state for user and service support and retrieves iFC from the HSS when necessary. When a session is being set up, iFC based invocations are propagated through a priority list of application servers. However, only sequential invocation (cascading) are supported by this process. It is not possible to switch dynamically between application servers. Once the service invocation has reached its terminating AS, it is no longer managed by the S-CSCF. It then becomes the sole responsibility of the terminating AS to cooperate with other application servers if this is necessary. Dynamic switching would be a great advantage for a composite service. The lack of dynamic switching limits the flexibility with which composite services can be used. In principle, all services that are to be used in a given session must be decided upon a-priori.

### 4.3.2 Logical Placement

The decision of where to implement the SB logic will naturally be influenced by the model being used but there are also important functional considerations. Three options have been suggested in [28]:

1. Collocated with S-CSCF entities.
2. As an independent functional element between the AS and the S-CSCF as illustrated in figure 4.1(a), or
3. Collocated with SIP AS entities, as illustrated in figure 4.1(b).

The next three sections discuss these options in greater detail.

### Collocated with S-CSCF

Recall from section 2.4 that the S-CSCF resides in the control layer and is already responsible for several central session management tasks for user and service endpoints, including the propagation of initial filter criteria. Recall also that the interface between the S-CSCF and the SIP AS supports only SIP.

Service chaining in IMS is static in nature. To improve IMS service interaction we need to add dynamic capabilities and flexibility in terms of what type of external services IMS can handle. We have established that we want to use the SB to implement this added functionality but if we in addition want to place the SB in the S-CSCF we may produce several new problems to the architecture. The potential problems we can identify are listed here:

- Placement of service functions in the control layer would violate the separation of roles in layered architectures.
- Adding WS capability to the S-CSCF requires additional reference points for HTTP protocol support.
- Adding complex service orchestration functions to the S-CSCF would significantly increase the load on the S-CSCF.

We can therefore already reject this option as a reasonable placement of the SB.

### Stand Alone

We have now established that it is not a good idea to put the SB in the control layer. Therefore, as a stand-alone element between the AS and the S-CSCF we assume placement in the service layer. This is a better solution but might not be sufficient for handling coordination of multiple services within the same AS. We therefore consider placement of such a stand-alone SB as a possible partial solution to a broader SB architecture.

### Collocated with SIP AS

Recall that the internal workings of the SIP AS is beyond the scope of the IMS specifications. This gives us an opportunity to define an AS that includes SB functionality without changing any of the existing interfaces or functionality existing in the IMS specification. It will also meet the necessary requirement of providing orchestration for services within one AS.

### 4.3.3 Constellation: Centralised, Distributed or Hybrid?

Here we consider three SB models that take into account possible placement of the SB as suggested above. Several constellations have been suggested for IMS service brokering architecture, they fall specifically under the categories of centralised, distributed or hybrid models. In the centralised model, one central service broker coordinates multiple application servers. This service broker would be transparent application servers,

the S-CSCF and the HSS. Based on our discussion in section 3.5 the centralised model could serve as a basis for a pure orchestration solution.

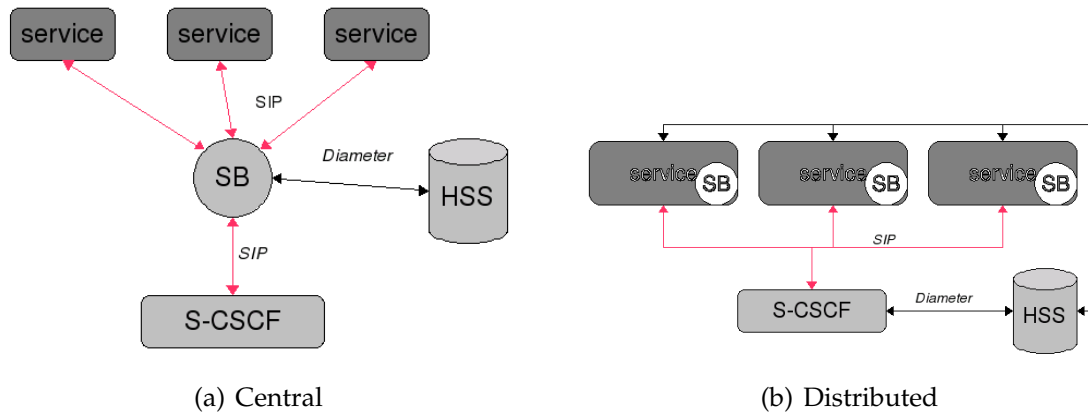


Figure 4.1: SCIM: Centred vs. Distributed

In the distributed model each AS has its own SB. Each SB can be embedded in the AS, as pictured in figure 4.1(b) or as its own independent modules outside the AS. Based on our discussion in section 3.5 the centralised model could be a basis for an orchestration solution while the distributed models could be a basis for a choreography solution. Each SB can communicate with the S-CSCF on behalf of their respective application servers.

As the name implies, hybrid models combine aspects of both distributed and centralised models. Some SBs serve the role of coordinating several application servers, others serve the role of collaborating with other SBs while yet others serve both roles. Figure 4.2 is one possible configuration. We suggest that given the potential level of orchestration complexity as well as the varied needs of composite services, it is not sufficient to deploy only centralised or only distributed SBs. We therefore suggest that some form of hybrid model be adopted.

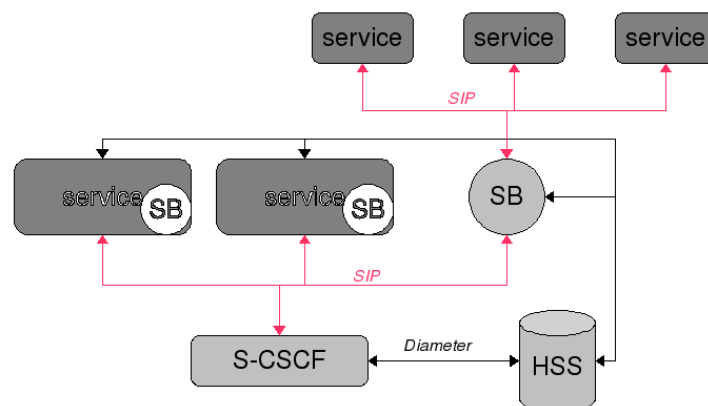


Figure 4.2: Hybrid SCIM

## 4.4 Orchestration Options

One key question must be addressed in order to choose the best orchestration options: At what level should orchestration be performed - via SIP sequencing, via WS interfaces, or as a combination of these? Figure 4.3 illustrates the different orchestration levels for service orchestration in IMS.

- As illustrated in level 1, orchestration by way of SIP sequencing gives the best runtime efficiency but the worst development efficiency. This level of orchestration relies on good design of iFC, to coordinate by the S-CSCF for use in standard SIP application servers. This approach, though it gives the best performance, gives the least flexibility. Application developers would be required to gain specialist knowledge of the IMS control layer, an inefficient development strategy and most likely an unsuccessful method for attracting new services to the IMS market. In addition, the limitation of static service chaining cannot be addressed if all orchestration continues to be performed at level 1.
- Level 2 involves implementing a proprietary service framework in a black box. As this solution provides no flexibility and no possibility to examine the architecture, it may, for our purposes, be rejected without further consideration.
- In levels 3 and 4 all development is done in the application layer with familiar development tools. This is the most efficient development approach but it may increase latency. The approach in this thesis is to concentrate on level 3, where careful design of service brokering framework with the help of, for example, SIP Servlets.

The disparity between the requirements of business processes and the requirements of Real Time services suggests that different levels of service orchestration need to be defined. Based on a discussion in [29] we introduce the notion of multi-level service orchestration. The current typical usage of SOA and Web Services is not sufficient for real time telecom services. However, some existing, WS applications are still relevant in IMS scenarios. A first layer of orchestration with strict latency constraints and a focus on interaction could be adopted as service orchestration level 1. Orchestration level 1 could then be used solely for real-time applications. Specific examples of services that would require level 1 Orchestration are Video conferencing and Push to Talk over Cellular (PoC). In other words, typical IMS services. A second class of WS orchestration could be classified as Service Orchestration Level 2. Orchestration level 1 would then be used for services that do not have real time requirements. Business processes and Web 2.0 services discussed in 3.9 would fit into this category.

## 4.5 Quality of Service (QoS) Aspects

In the process of designing a service orchestration framework, QoS aspects must be considered. A thorough QoS study of a distributed architecture would take several years. However we can start by identifying availability and latency as two key aspects. We consider these aspects in the particular context of composite services with modules from different service providers and identify challenges and requirements.

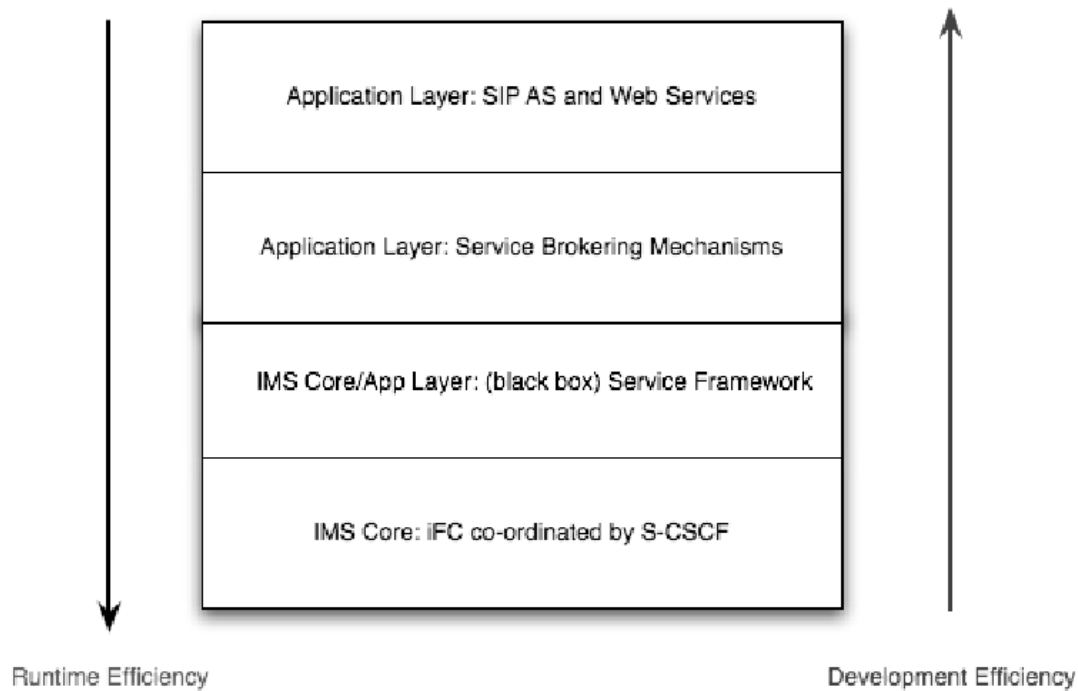


Figure 4.3: Orchestration Levels

### 4.5.1 Availability

Availability is a necessary topic of study when considering distributed systems. The availability metric is usually defined as the percentage of time a system is operational in the course of its lifetime. By this traditional understanding of availability in distributed systems, availability is usually given as an atomic measure: at time  $t$  a system is either available or not available. However, a composite services such as those we discuss here introduce additional considerations. The loosely coupled service modules may be from different providers, possibly even based on different protocols (SIP and HTTP for example) and with different performance expectations. This raises some questions: What if the IMS delivered voice component of a composite service is available but the web based presence component is not? Is this composite service available, partially available or unavailable? How do we quantify the level of availability contributed by each service part? To address such issues a modular availability model is presented in [27] for more details on this topic.

### 4.5.2 Session Setup Latency

An analysis in [30] measured the time used to initiate and terminate a voice session between two registered IMS user end-points using a Focus IMS core<sup>1</sup> as well as the percentage of retransmitted messages that were sent. The test did not include any service invocation. The delay was found to be from 0.013 seconds over fast ethernet

<sup>1</sup><http://www.openimscore.org/>

networks and 0.014 over 802.11g wireless networks and 0.159 seconds over 802.11b wireless networks. There were no retransmissions over fast ethernet or 802.11g while 802.11b had 5.4% retransmission.

### 4.5.3 SIP Proxy Latency

According to [30] routing of SIP messages consumes a large percentage of processing power in IMS networks. It is therefore very important to make SIP proxy entities in IMS as efficient as possible. Recall from 2 that the SIP AS (and by extension any SIP service broker entities in the service layer) operate in various proxy server modes. In [31] Nahum et.al measures the throughput and response times of SIP proxy servers given different load levels. The most significant findings were that (1) Authentication procedures can reduce performance by as much as 90%, (2) Using TCP instead of UDP can reduce performance by as much as 65% and (3) Enabling stateful processing can reduce performance by as much as 60%. Response time for overloaded servers increased exponentially. Other jumps in response time are directly due to the exponential back-off algorithm used by SIP's transmission timers (for example timers A, B and D depicted in the state machine in figure 3.2). Let us consider each of these performance degraders:

#### Authentication

In practical terms, an IMS subscriber using her terminal in a typical manner could be expected to register approximately once a day and to have registration included as an automatic feature (for example on turning on the telephone or on local log-on to her computer). It is at this time that the iFC are downloaded to the S-CSCF and also at this time that HTTP Digest Authentication methods are used. Since we want to concentrate on the actual service composition interactions we will assume for our purposes that the user is registered and that no service profile updates have occurred since registration. In so doing, we can disregard the performance reduction involved in authentication.

#### TCP

According to the specification, all SIP elements are required to implement both UDP and TCP (they may implement other network protocols as well). UDP's statelessness and simplicity make it the preferred protocol for telecom applications that require, respectively, high throughput of short messages and low latency for each message that is sent. However, requiring TCP support ensures that received messages can be processed even if they are too large to be handled by UDP. Maximum datagram packet size for UDP is 65,535 bytes, including IP and UDP headers. For the SIP protocol, message size limits for sending over UDP are either (a) at least 200 bytes less than path Maximum Transmission Unit (MTU), or (b) less than 1300 bytes if the path MTU is unknown (based on an assumed 1500 byte Ethernet MTU). Above those limits, TCP or another congestion controlled transport protocol must be used. Knowing the performance penalty for TCP, we now set a design requirement that messages sent over our orchestration plane must be small enough to be handled by UDP transmission.

However it must be noted that sending SIP over TCP is be more secure. Although security issues are not handled in depth in this thesis, we propose that security in IMS is best handled by other mechanisms like the Diameter AAA routines, SLA and HTTP authentication, leaving SIP as lightweight as possible to concentrate on the task of efficient session handling.

### **Stateful Processing**

A stateless SIP proxy is the only SIP entity that does not use the transaction layer. It simply forwards messages without having any knowledge of whether it original or a retransmission and without providing its own provisional messages. If the address in the via header does not match the stateless proxy to which it was sent, the message is silently discarded. Using only stateless proxies as service brokers implies that the intelligence would have to be placed into other IMS entities along the service chain. This could result in overloading the S-CSCF or excessive messaging between the AS and the HSS. These options are likely to cause an even greater performance degradation. Therefore, stateful proxies should be used whenever possible without adversely affecting the load imposed on other nodes. One area in which delay can be reduced for stateful proxies is by not making transaction-stateful proxies call-stateful in addition. Call-stateful proxies retain dialog state from the initiating INVITE request to the terminating BYE request - thus introducing even more performance overhead.

### **Load Balancing**

Standard load balancing techniques (replication, backup, multiple core processing) should be able to provide sufficient dimensioning of SIP nodes. Commercial suppliers provide capacity specifications for their products. At time of writing this included:

### **Transmission Timers**

The delays caused by transmission timers are built into the SIP protocol [12] so they must be taken into consideration. The timers introduce an element of reliability to SIP when messages are transported over unreliable protocols such as UDP. Estimated Round Trip Time (RTT) is the base value for SIP timers. It defaults to 500ms and is set to 600 ms in our testing. Table 4.1 gives an overview of the most important SIP timers. High-latency transactions can trigger retransmission timers which, by design, trigger increased delays with each iteration. So the challenge is to reduce retransmission induced latency by in-effect keeping latency inherently low.

The 200 byte margin between the message size and the MTU allows for receiving SIP responses that are larger than corresponding SIP requests. Approximately 30 bytes are used by IP/UDP (assuming no IPSec). 1300 is chosen when path MTU is not known, based on the

## **4.5.4 Web Services Latency**

The findings reported in [32] support our theory about SOAP's impact on Web Services. Davis and Parashar tested various implementations of SOAP and compared

Timer	Purpose	Value	Note
A	request retransmission	estimated RTT	mandatory over UDP <sup>2</sup> . Value doubles with each retransmission.
B	transmission time out	64 x RTT	Mandatory. Represents time to send 7 requests over UDP
C	flexible timeout handling for proxied requests,	> 3 minutes	Mandatory for proxied INVITE requests in client transactions
D	transaction completion	>= 32 seconds for UDP	Optional. Represents time that the server transaction can remain in the "Completed" state for UDP

Table 4.1: SIP Transmission Timers

them to Java RMI and CORBA. SOAP processing was found to use 80% of the time used in a SOAP call that does nothing and the Microsoft SOAP kit was found to use 94% of the time on the same call. They concluded that the overhead in SOAP messaging was largely due to time used to parse and format XML as well as the fact that multiple system calls were sent for the transfer of one logical message. Note that this analysis was done on Simple Object Access Protocol (SOAP) 1.1. The current version is SOAP 1.2 (no longer an acronym). However, the findings are still relevant.

In [3] Lynch performed informal latency benchmark tests on the same WS code in SOAP ColdFusion<sup>3</sup> Components (CFC) and REST ColdFusion Markup (CFM) versions. Trials were done for Cold Fusion versions 7 and 8 and with and without caching. Although it not considered an official benchmark the results are too dramatic to be ignored. According to the published statistics from these tests the REST versions of the code ran 4 to 5 times faster than the SOAP version. Figure 4.4 is a graphical presentation of the results.

## 4.6 Problem Summary

The challenges of service orchestration in IMS can be summarised as follows:

- The static nature of existing iFC chaining limits the flexibility with which IMS services may be composed. It also challenges the integration of some popular third party web services that have already gained popularity among potential IMS customers. In particular, services that work well in web 2.0 scenarios do not conducive to static service chaining.
- On the other hand, IMS is a multimedia communication framework and the real time requirements must be met for voice and streaming. For this, traditional WS architecture is insufficient for several reasons:

<sup>3</sup>An application server from Adobe - popular with WS developers

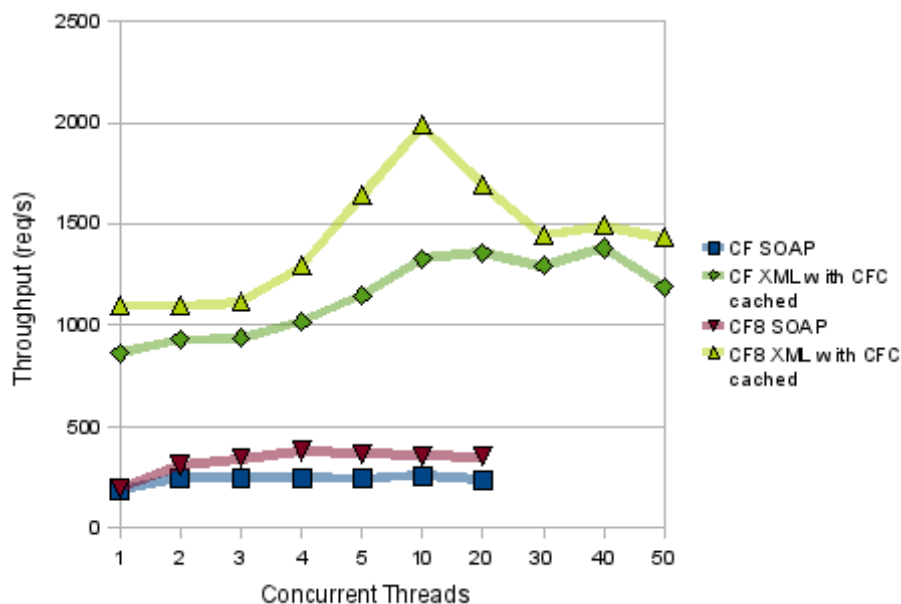


Figure 4.4: Web Services Throughput: SOAP vs REST [3]

- Average latency is too high and there is no latency guarantee
- Reliability is too low and there is no reliability guarantee
- SOAP messages are too verbose. This point in itself adds to the latency issue.
- Security is not a part of the default WS standard which means that exposing an application as a WS makes it very vulnerable to non-secure 3rd party WS elements
- Atomicity, Consistency, Isolation, Durability(ACID) qualities are not upheld.

Although the OSA CSC provides a gateway to external telecom WS, it does not provide a solution for native IMS services.

## 4.7 Hypothesis

From our discussion so far we can establish that a suitable framework for Service Orchestration in IMS must support quadruple play communications services that IMS was designed for. There is also a need to incorporate third party web services into the orchestration framework. These two classes of services have different needs in terms of latency and availability. Therefore, we need an orchestration framework that can support the the external non-real time services without compromising the native IMS real-time services. To do this we need to be able to apply a composite availability model to the composite services. This leads us further to the following hypothesis:

A suitable orchestration can be implemented as a hybrid Service Oriented Architecture (SOA). To isolate the different types of services from affecting each others performance, at least two different layers of orchestration should be incorporated: One for

native SIP service chaining and one for dynamic discovery and incorporation of third party WS. In an effort to minimise the known issue of WS message overhead, REST should be used instead of SOAP.

# Chapter 5

## Model

In this chapter we construct a model of the service orchestration architecture that we would like to prototype. We start in section 5.1 by presenting interfaces of the IMS entities that are directly involved in service provisioning. This builds on the introductory material presented in chapter 2. Section 5.2 presents details of the component classes in the user and service profiles. Here we will see which data types and data structures we have to work with. Recall from 2 that user specific details for service provisioning are stored in the service profile in the HSS. Section 5.3 then outlines the signal path for service provisioning in IMS. Specifically it presents the basic steps for adding IMS services to a SIP session that is being set up between two IMS subscribers. Section 5.4 highlights the options we have for incorporating web services into the IMS domain and especially the options for incorporating 3rd party WS elements. Section 5.6 introduces the notion of an orchestrating SIP AS that called a SIP Service Domain and discussed alternatives for its implementation. Section 5.7 discusses development platforms and APIs that could be used to develop such a SIP AS.

### 5.1 Existing Service Creation Interfaces

There are several ways for the key components SIP AS, UE, S-CSCF and HSS to communicate with each other. This gives us flexibility in our design process. Figure 5.1 presents the existing interfaces we have at our disposal. We can use the XML Configuration Access Protocol (XCAP) over the Ut reference point. This protocol is based on HTTP and is thus compatible with the IMS specification. XCAP provides an opportunity for external users to expose data to an AS in an IMS domain. This data will typically be information that can only be set by the user, like adding a device to the Presence resource list or updating static Presence information. Using XCAP to incorporate user-generated information adds a method for making an IMS based architecture Web 2.0 friendly.

Another important quality of the HTTP interface is that it is the only interface needed to communicate with external web services. Recall from section 3.4 that all the WS protocols operate over HTTP. Based on our discussions in 3.4 and 4.5 and our hypothesis as presented in 4.7 we discard SOAP as a part of our WS platform. What we are left with is pure HTTP requests and responses as well as XML data structures. Our prototype includes only one HTTP service, a web portal. Given that it is a portal,

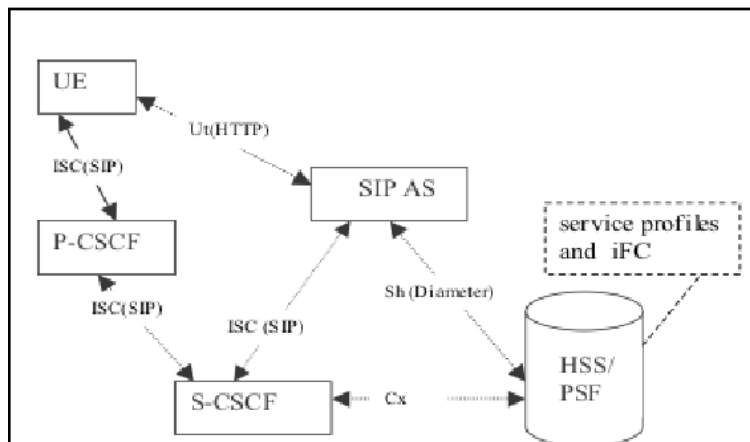


Figure 5.1: Service Creation Interfaces

its role is more that of a service gatherer/presenter/interface for the IMS user, rather than that of a web service. Therefore there is no focus on exposing web services but rather on consuming them. However, the design does allow for the IMS service layer to provide its own web services and in that case WSDL documents would have to be offered. For our current scenario, the SIP AS needs only to be able to read and understand WSDL documents from external services.

## 5.2 IMS Service Routing

The S-CSCF receives an initial SIP request from a UE via a P-CSCF. Communication between the UE and the S-CSCF may also pass through the I-CSCF. This may happen, for example, if the message originates from a foreign domain. In this case the I-CSCF serves to anonymize the servers in the home network to protect them from untrusted domains. If necessary, the S-CSCF retrieves an updated service profile from the HSS. The service profile includes iFC if there are any. To simplify our implementation, all our scenarios assume that the user is already registered and that the service profile that was retrieved at registration is up to date.

An IMS user profile consists of a subscription and at least one Service Profile as illustrated in figure 5.2(a). The details of a service profile are illustrated in figure 5.2(b)

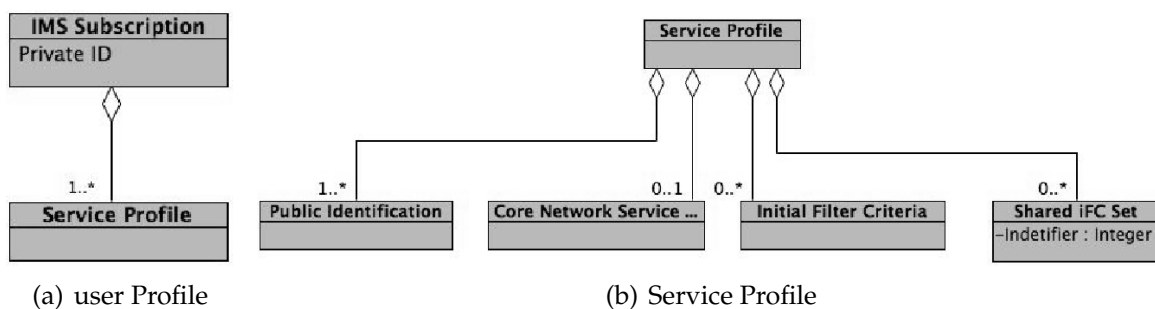


Figure 5.2: Relationship between User and Service Profiles

Assume that iFC are present. Based on the iFC, the request is sequentially routed to specified prioritised list of application servers as defined in the iFC class in figure 5.3(a). Trigger Point is a Conjunctive or Disjunctive Normal Form (CNF/DNF) Boolean expression of Service Point Triggers.

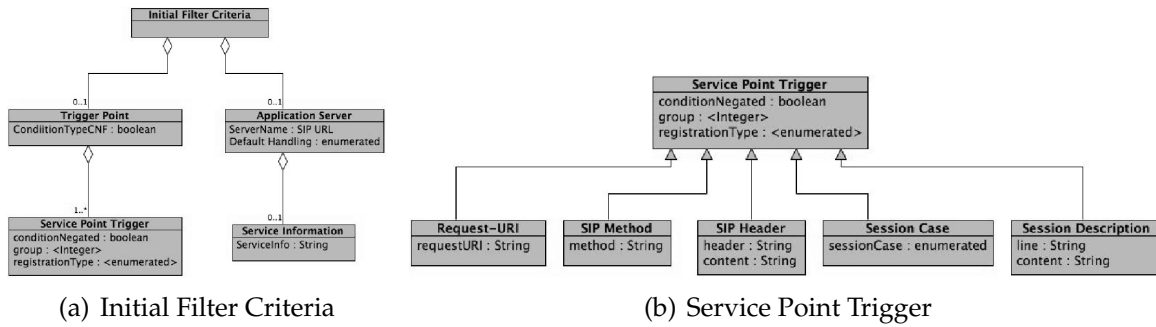


Figure 5.3: Relationship between iFC and SPT

There are five defined Service Point Triggers (SPT) on which iFC trigger points can be based [33]. Each SPT is indicated with a Boolean value that can then be used as part of the CNF/DNF expression in trigger Point. The SPT are illustrated in the class diagram in figure 5.3(b) and explained in the following list:

- Request-URI: The target address of the request
- SIP Method: INVITE/SUBSCRIBE/MESSAGE etc as defined in the IETF SIP specification.
- Sip Header: The presence or absence of any header field (including unknown fields) and their contents.
- Session Case: The direction of the SIP request that can be one of a total of four possibilities; UE-originating or UE-terminating for either registered or unregistered users.
- Session Description: This can be used for session-content negotiation

These details of IMS application trigger behaviour will be leveraged in our orchestration design. Figure 5.4 outlines the series of objects that are invoked in order to pass the message on to the correct Service Broker when Alice initiates a SIP request. In the iFC model as depicted in figure 5.3(a) the Service Information attribute contains an optional string. We can use this string to transparently pass extra information on to the application servers.

When an end user registers, service information, if present, is passed transparently from the S-CSCF (S-CSCF must be in UAC mode in this case) to the AS indicated in the iFC [33]. The transparent nature of the service information implies that we can use it to pass information to

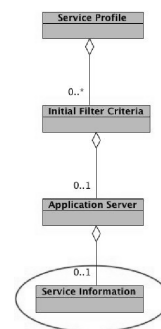


Figure 5.4: AS Invocation

services without adding any processing overhead to the S-CSCF. The S-CSCF can then remain free to concentrate on coarse-grained service chaining as in its original design. It will ignore the service information, which we could then use to parse data indicating which higher-level service brokers need to be invoked.

### 5.3 Service Provisioning and Request Routing

Let us illustrate the concept of service chaining by way of a scenario. Generally, when the IMS core receives an originating SIP request from a UE, the identity of the target party/parties is resolved to a specific target or group of targets. If there are services listed in the senders service profile then these services must be invoked before the request is routed further towards its target. If there are also services listed in the target's service profile then these also have to be invoked before the request reaches the target. Figure 5.5 illustrates such an example of session initiation with service chaining between two IMS end users from two distinct IMS home domains. Assume that end users Alice and Bob are already registered with their respective IMS home domains. This assumption implies that:

- Alice and Bob have already each been assigned a specific S-CSCF in their home domain.
- Let  $S\text{-CSCF}_A$  be the S-CSCF that has been assigned to serve Alice.
- Let  $S\text{-CSCF}_B$  be the S-CSCF that has been assigned to serve Bob.
- $S\text{-CSCF}_A$  and  $S\text{-CSCF}_B$  have already retrieved Alice's and Bob's service profiles from the each respective HSS.

Let us also assume that there have been no updates to Alice and Bob's user profile since registration. This assumption implies that the S-CSCF does not need to retrieve any updates from the HSS when handling the request. The following conditions also apply to this scenario:

- HNA is Alice's home network.
- VNA is the visited network in which Alice is currently located.
- HNB is Bob's home network.
- VNB is the visited network in which Bob is currently located.

Alice selects Bob's name from her address book list on her mobile phone. The numbered sequence of events as shown in figure 5.5 are as follows:

1. The IMS client on Alice's phone sends a SIP INVITE request addressed to Bob.  $P\text{-CSCF}_{VNA}$  is discovered as the IMS entry point (by for example DHCP)
2.  $P\text{-CSCF}_{VNA}$  routes Alice's request to  $S\text{-CSCF}_A$ . The prioritised list of initial filter criteria (iFC) in Alice's service profile indicate that  $AS_1$  (Voice Mail Box), is the first service to be invoked.  $AS_1$  does not terminate the request so control is returned to the S-CSCF.

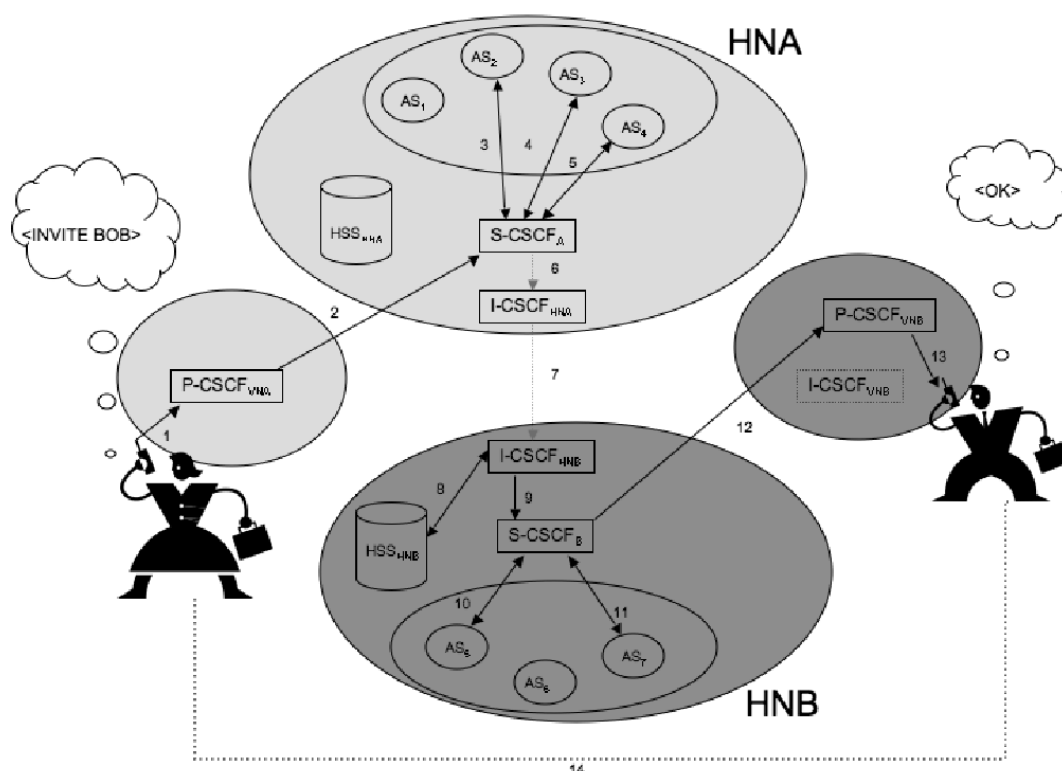


Figure 5.5: Generic Request Routing

3. The S-CSCF invokes the next service in the list,  $AS_2$  (Video Streaming Server).  $AS_2$  does not terminate the request so the control is returned to the S-CSCF.
4. The S-CSCF invokes the next service in the list,  $AS_3$  (Presence).  $AS_3$  does not terminate the request so the call is returned to the S-CSCF.
5. There are no more services to invoke so  $S-CSCF_A$  examines the request URI and discovers that Bob's home network is an untrusted foreign network HNB, so the request is routed via  $I-CSCF_{HNA}$  (which serves a masking function here)
6.  $I-CSCF_{HNB}$  accepts the incoming message examines the header and,
7. References Bob's user profile in  $HSS_{HNB}$  to determine which S-CSCF serves Bob.
8. It can then route the request to  $S-CSCF_B$ .
9. The first service in Bob's iFC,  $AS_5$  (Multimedia Server) is invoked and control is returned to the  $S-CSCF_B$ .
10. The second and last service in Bob's iFC,  $AS_6$  (Presence Server) is invoked and control is returned to the  $S-CSCF_B$ .
11.  $S-CSCF_B$  routes the request to  $P-CSCF_{VNB}$ .
12.  $P-CSCF_{VNB}$  routes the request to Bob's phone.

13. Bob accepts the request (the steps involved in accepting are not shown).
14. The Session Description Protocol in the headers from both Alice and Bob indicate that each have video streaming capabilities. A new video call session is set up between Alice and Bob (the steps in service capability negotiation and request reservation are not shown). Based on the services that were triggered in this session set-up, both Alice and Bob's Presence information will show that they are busy and Alice's voice mail server will handle all incoming calls while Alice is talking to Bob.

This scenario illustrates the basic intention of IMS. In our experimental set-up, a subset of this scenario will be expanded to show how service brokering can be used to:

1. Make service brokering dynamic; and
2. Include external non-IMS services in the service chain.

## 5.4 Incorporating Web Services

Here we consider two possibilities of converging IMS services and Web Services with the collaboration of third party service providers: With control in the 3 party domain or with control in the IMS domain.

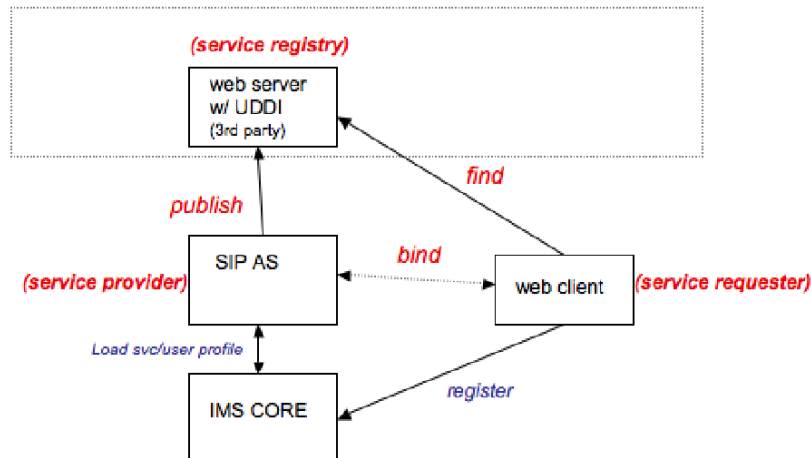


Figure 5.6: WS/IMS Scenario: Control in Foreign Domain

In the first set-up, as illustrated in figure 5.6 the SIP AS in the IMS domain is a WS service provider and publishes its services to a service registry residing in a foreign domain. Service requesters, which could include IMS UE, would then consult the registry to discover IMS services. This option puts control in the foreign domain.

In the second set up, as illustrated in figure 5.7, the SIP AS performs in B2B UA mode and discovers external services on behalf of IMS clients. The IMS UE needs only to relate to the SIP AS. Here control is kept in the IMS domain. This suggests

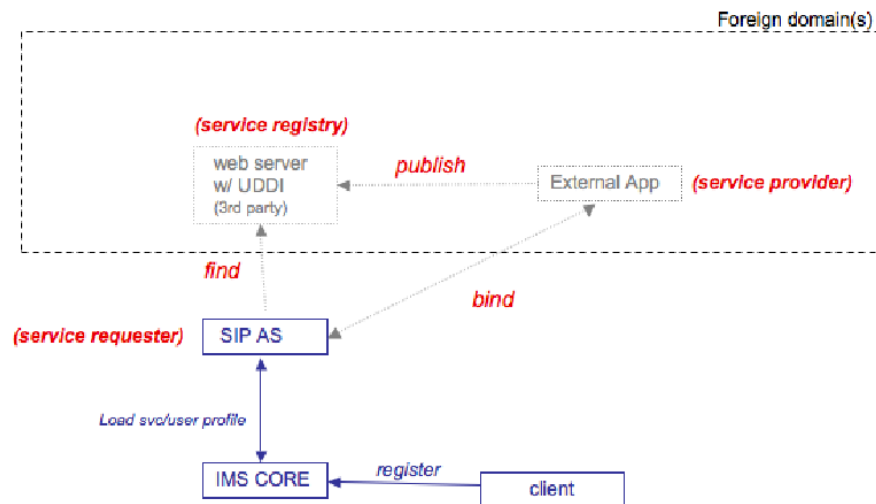


Figure 5.7: Control in IMS Home Domain

less configuration overhead for the UE as well as better security mechanisms towards foreign domains. This is a more attractive option for the IMS operator and thus the option that will be explored further.

The next step is to develop a service broker architecture that supports WS convergence, while keeping control in the IMS domain and diverging much of the overhead from the SIP client to the SIP AS. To do this a specific type of SIP AS is introduced.

## 5.5 SIP Service Domain

Here the notion of an extended SIP AS is defined. It is extended in that it fills the basic requirements for a SIP AS. This extended AS will be called a SIP Service Domain (SSD). This SIP AS conforms to the IMS standard in so much that it performs standard communication mechanism with the S-CSCF using SIP, with the HSS using Diameter and with UEs over an HTTP based protocol, XCAP. In addition, the SSD has a hybrid set up of:

- Multiple service modules,
- A configuration of service brokers with varying degrees of advanced functionality,
- An HTTP interface that is also used for communicating with remote 3rd party web services.
- A gateway (that may or may not be an integrated service broker function)

## 5.6 SSD Service Broker Architecture

Services are assigned to brokers according to service type, how frequently they are chained together and whether they are local or remote. Care is taken to ensure that the



## Level 2/Gateway

Each SSD has 0 to 1 level 2 SBs (gateway broker). A gateway broker's main task is to communicate with remote 3rd party service modules. The interface from this kind of SB must support, in addition to SIP, RESTful HTTP, SOAP and RPC as well as security and authentication mechanisms.

### 5.6.2 SB Proxy or SB Back-to-Back UA

Given the decision to implement the service broker as a SIP Servlet we need to decide whether to implement it as a back-to-back user agent (B2BUA) or as a proxy.

#### B2BUA

As a B2BUA the SB wraps initial incoming client-originated requests into new requests before sending it to the requested service. All subsequent requests/responses from the service to the client would be handled by the SB. This gives more granular control of service behavior but also more overhead on the SB. For high traffic periods such overhead could produce a noticeable degradation in response time.

#### Proxy

Implementing the SB as a proxy provides the opportunity to implement it as stateless SIP entity whereby messages are simply forwarded without knowledge of context. This method be in keeping with REST architectural principles. It would also mean that the service level intelligence including state information would have to be the responsibility of the clients. This is in keeping with the original intentions behind the design of SIP.

## 5.7 Choice of Service Delivery Platform

Based on discussions rooted in Khlifi and Gregoire's comparison [5] we consider SLEE and SIP Servlets as two possible foundations for our SDP.

### 5.7.1 Mobicents Implementation of JSLEE

Mobicents [2] offers a development environment for several different types of applications. As figure 5.9<sup>1</sup> illustrates, our SIP application, presence server, web portal and media server could all have been deployed in a single Mobicents environment. At time of writing the most current specification was JSLEE 1.1 (JSR 140), approved in 2007 but Mobicents has not yet moved towards JSLEE 1.1 compliance.

Mobicents is, to the best of our knowledge, the only freely available, open source JSLEE implementation to date. Mobicents 1.2.0.BETA2 also includes Sip Servlets 1.1 (JSR 289) support. It is as a Voice over IP (VoIP) middleware platform and therefore

---

<sup>1</sup>[code.google.com/p/mobicents/](http://code.google.com/p/mobicents/), copyright 2008 Google

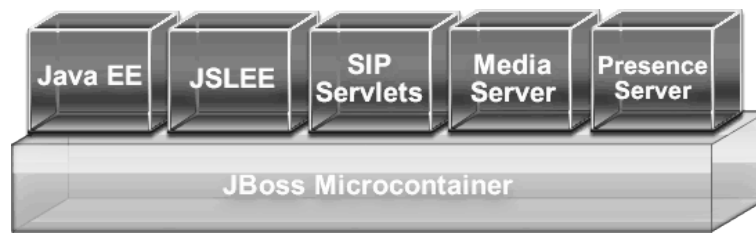


Figure 5.9: Mobicents

designed for low latency, high throughput applications. Mobicents SIP services can be deployed in one of two modes: (1) SIP proxies responding to invite requests and (2) everything else. SIP services rely on the Mobicents SIP resource adapter. The Mobicents platform includes full JSR 289 support.

Another JSLEE implementation on the market is the Rhino Real Time Application Server from Open Cloud<sup>2</sup>. Rhino sometimes markets this product as an IMS application server. The latest release (June 2008) is advertised as SLEE 1.1 compliant. Open Cloud has played a central role in developing both JSLEE 1.0 and JSLEE 1.1.

### 5.7.2 Sailfin's Implementation of SIP Servlets

Sailfin from Sun Microsystems is a SIP Servlets framework implemented on top of Sun's J2EE server Glassfish container. Sailfin has partial JSR 289 support (the AR is not yet implemented).

---

<sup>2</sup>[www.opencloud.com](http://www.opencloud.com)

# Chapter 6

## Implementation Details

This chapter provides details of the SSD implementation and context. First section 6.1 presents the implementation scenario. Section 6.2 describes the IMS core implementation that was used. Section 6.3 describes the clients that were used. Finally section 6.4 describes the SSD model that was implemented.

### 6.1 Scenario

In this section we apply the generic models and specifications to a concrete scenario. The scenario covers a basic IMS case in which both IMS users are already registered and both users are being served by the same S-CSCF. However, the scenario also introduces communication with non-IMS, non-SIP entities, namely Facebook, Flickr and del.icio.us Web Services. The main actor is Alice - a customer with an IMS subscription who was already an active user of Facebook and Flickr before signing up for an IMS subscription. The services that Alice uses for personal email, instant messaging as well as fixed and mobile telephony are all managed through her IMS subscription. It is Alice's Service profile that determine which services will be invoked.

#### 6.1.1 Default Service Profile

The iFC in Alice's default service profile indicates that the presence service should be activated whenever Alice is registered. When Alice registers, a SIP PUBLISH method is automatically sent to a presence server in the IMS domain indicating Alice's status. The default status is "available" but Alice can change it to a custom status such as "do not disturb". The Presence Server then notifies all the users on Alice's buddy list that Alice is online.

#### 6.1.2 Home Service Profile

We consider a simplified version of a user profile. Alice's subscription contains 1 service profile, called `home_sp` which is depicted in figure 6.1. The service profile has 1 public service identity - which is a SIP URI `alice@dus04.nta.no`. It is not a temporary

identity, so the barring ID is unset. The profile does not contain a Core Network Services Authorization class, therefore, Alice has no restrictions on media types she can use or available services she is allowed to access.

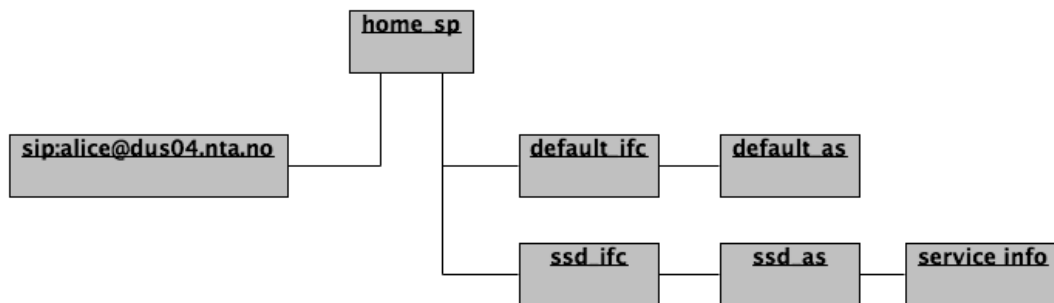


Figure 6.1: Alice's Home Service Profile

### 6.1.3 Services

There are three groups of services available to Alice. They are as follows:

#### Instant Messaging, Chat and Voice Call

Assuming that she is already registered, the most basic use cases for Alice are (1) Placing a call to another registered IMS user or (2) Starting a chat session with another registered IMS user. From her buddylist, Alice selects a user who is indicated to be available. She then either (a) Initiates a chat session, (b) Initiates a voice call or c) Sends an instant message. Figure 6.2 is a snapshot of Alice's UE after a call is established with her IMS buddy Bob.

#### IP TV and Video Calls

Alice can also watch IP TV on her video enabled IMS client as well as initiate and receive video calls with other users that have video capabilities on their IMS clients. To do so Alice has to use a service profile that triggers the Multimedia server.

#### Internal and External Presence Elements

Alice can publish and subscribe to IMS friends on her local buddy list. She can also import her external buddy lists from Facebook, Flickr and del.icio.us into her IMS environment. To do so the SSD service must be specified in her iFC and the PSI of for "external buddies" resource list must be addressed in her invite request. The request will then be cascaded to the level 2 SSB which will in turn wrap the SIP method into an HTTP request and invoke web service calls to Facebook and Flickr.

### 6.1.4 Use Cases

Figure 6.3 depicts the use cases based on the services listed above. The use cases are described in the following list:

## 6.2 IMS Core Component

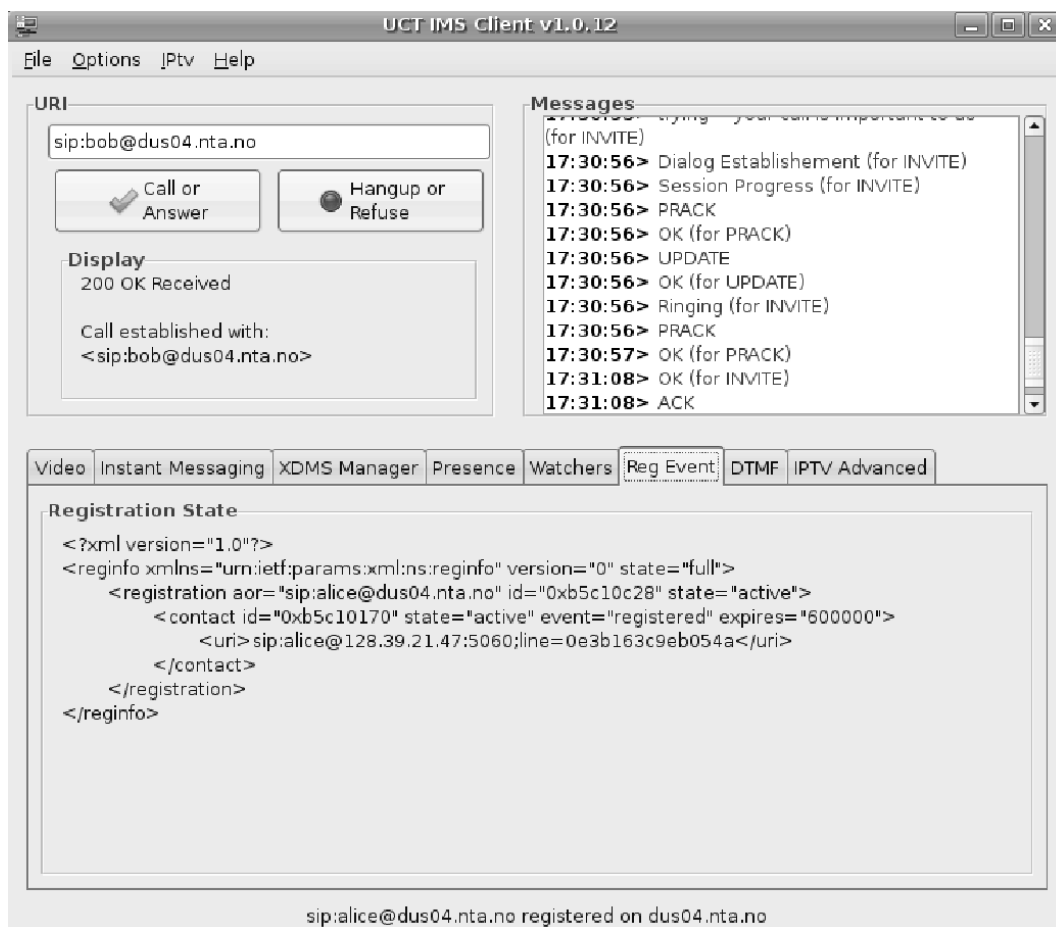


Figure 6.2: Call Established

The SSD was tested against a Franhauser Fokus Open IMS Core installation. Franhauser Fokus Open IMS Core is a freely available, open source (GPL v2) IMS testbed written in C. It implements the basic IMS core components (most notably all three CSCF and the HSS) and all the mandatory reference points to communicate with these components according to the IMS specification.

Like all research implementations the focus testbed is an evolving work but it seems to be fairly stable and of high quality and was very useful for testing the model. More details of Fokus' IMS core is available at [www.openimscore.org](http://www.openimscore.org).

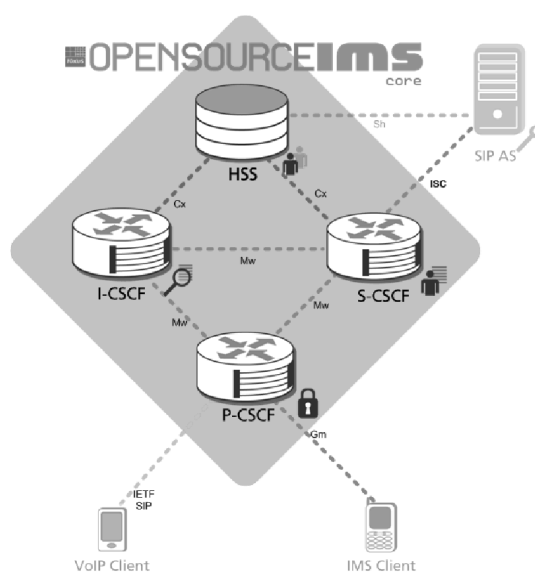


Figure 6.4: Fokus Open IMS Core [4]

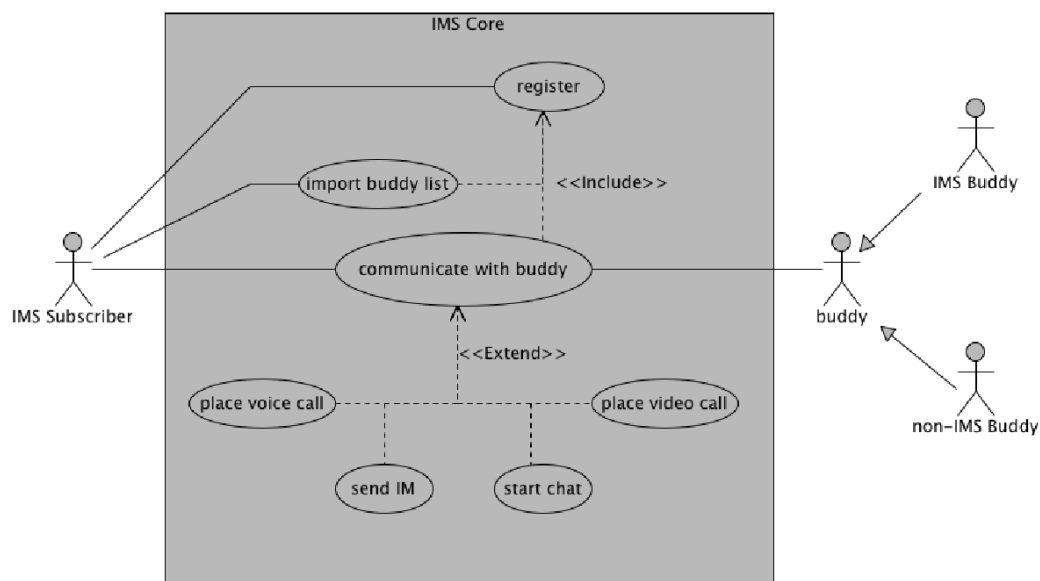


Figure 6.3: Use Cases

## 6.3 Client Implementation

The UCT IMS Client is an open source (GPL v3) client from the Communications Research Group at the University of Cape Town, South Africa. It was designed to work with the Fokus IMS core. It is one of the extremely few IMS capable clients that are freely available. There are several freely available IETF SIP clients but, without IMS support, they are not sufficient for testing with an IMS core. As is typical for small open source projects, the UCT client is a work in progress and does have some bugs and omissions, but at the time of testing it did have at least partial implementation of several of the basic elements that were useful for testing the SSD model. This included Authentication and Key Agreement (AKA) authentication, Presence, voice, video, IPTV viewer and XCAP client.

## 6.4 SSD System Model

This section highlights some design choices for the SSD prototype.

### 6.4.1 SSD Components and Deployment

The SSD is a SIP AS with hierarchical service brokers. It implements the required IMS reference points towards the UE, the HSS and the S-CSCF. In addition it extends the use of the HTTP and SIP interfaces to enable dynamic service composition as well the inclusion of external web services. Figure 6.5 shows the SSD components as (in blue), as well as the core IMS components (in pink) and the external web services (in yellow) that interact with them.

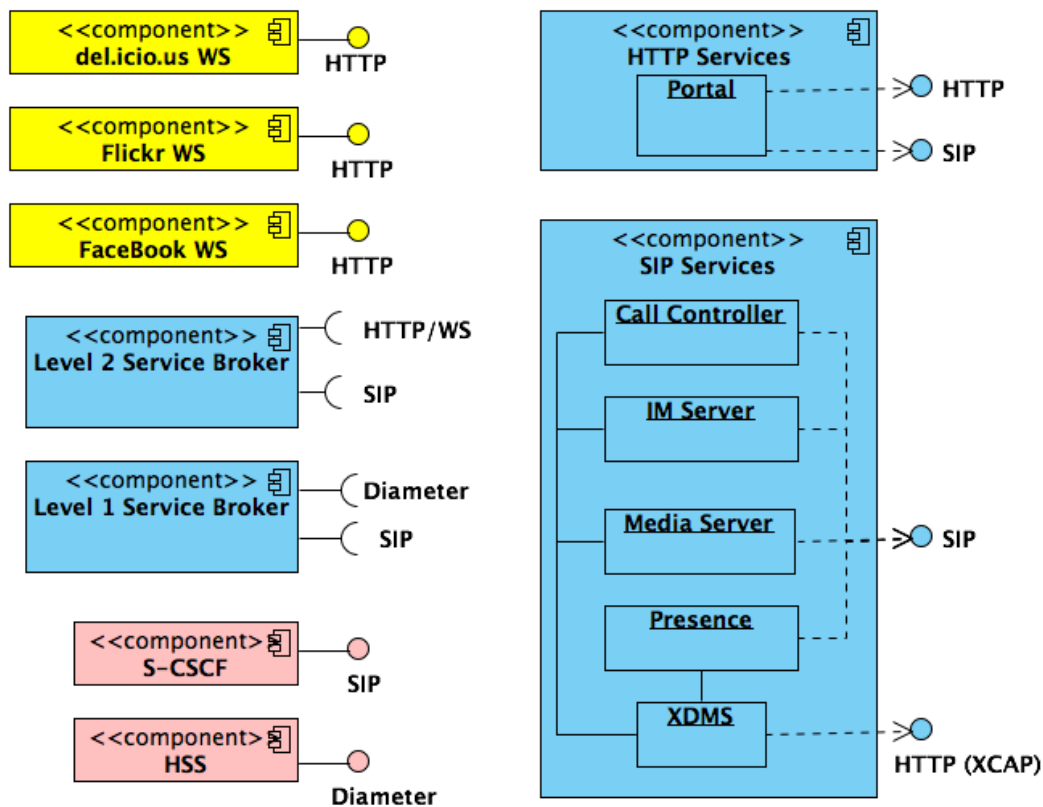


Figure 6.5: SSD (and surroundings) Component Diagram

## 6.4.2 Presence Service

A Colibria<sup>1</sup> server installation was used for the Presence server and the XDMS. Colibria Implements the OMA Presence SIMPLE enabler. The main specification used by this enabler is Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions (SIMPLE) from IETF and XML Document Management Server (XDMS) from OMA. Figure 6.6 gives an overview of the OMA SIMPLE architecture which is implemented on the server. Figure 6.7 show the interactions involved in a basic Presence scenario in our IMS environment wherein one authorised Watcher (Ola) subscribes to Presence updates from one Presentity (Kari). A generic description of the signal flow is as follows:

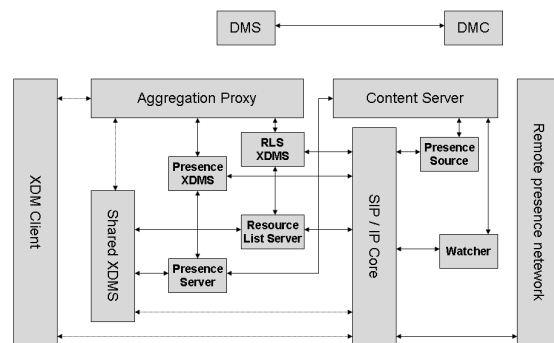


Figure 6.6: OMA Presence SIMPLE (wikipedia.org)

<sup>1</sup><http://colibria.com/products/presence-server-modules>

1. Presence source publishes to S-CSCF
2. S-CSCF publishes to Presence server
3. Presence server sends OK to S-CSCF
4. S-CSCF sends OK to Presence source
5. Watcher subscribes (to a presence source)
6. Presence server carries out authorization
7. Presence server notifies watcher of latest published update from presence source

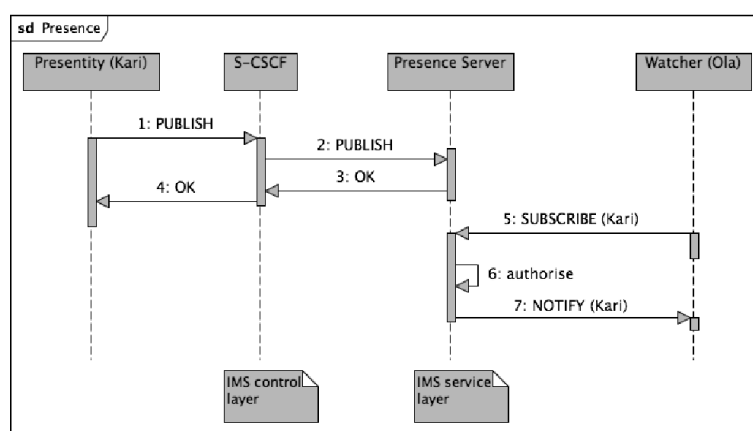


Figure 6.7: SSD (and surroundings) Component Diagram

The basic presence sequence in figure 6.7 involves only native IMS servers. It uses SIP to send and receive requests and the authentication phase can make use of Presence lists that are uploaded to the XDMS by the Presentities. Figure 6.8 illustrates the interactions involved in importing third party Presence information from Web Services. This sequence may occur in addition to the local presence sequence. Requests for external presence items are passed from SIP Servlets to HTTP Servlets before being passed on to the external WS as RESTful WS requests. The responses are sent to a web portal in the IMS domain to be consumed by the IMS client.

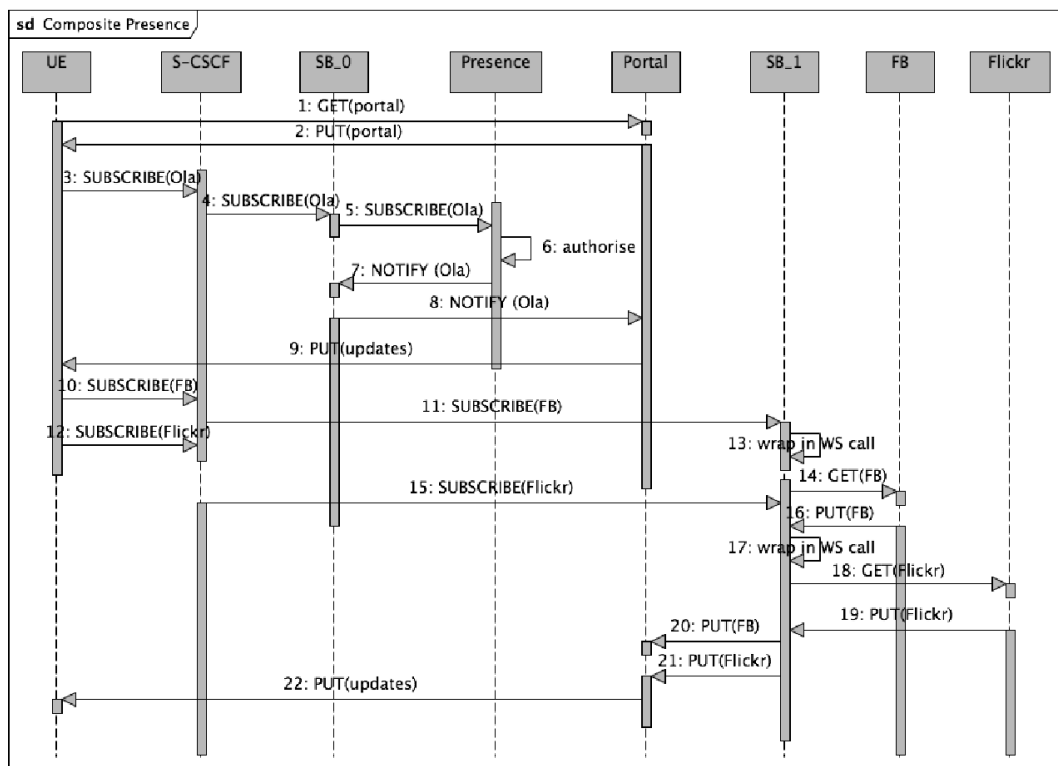


Figure 6.8: Composite Presence in SSD Sequence Diagram

# Chapter 7

## Experiment and Results

In the previous chapter we presented implementation details of the SIP Service Domain prototype as well our testing environment and scenario. In this chapter we present observations and results from trials done in our testing environment.

### 7.1 Availability

There was no observable downtime from the external Web Services during the experiments. This can probably be attributed to the fact that we used very popular Web Services that are used to heavy traffic. It is therefore expected that they have good redundancy and backup systems that would make it difficult for the user to notice node failures. It is further expected that a study of the same services over a long period (for example one year) would provide us with some failure statistics. We have already established that we cannot do a quantitative study on availability in the time available. What we were most interested in, in terms of availability, was whether it appears feasible to have an orchestration architecture that is modular enough for our availability model to be applied. It was observed that when incorrect information was deliberately sent to the remote Web Services the lack of response did not affect the other services adversely. Both the other Web Services and the SIP services were still able to perform correctly.

### 7.2 Latency

Latency measurements were recorded by tracing packets from session activity using the Wireshark Protocol Analyzer<sup>1</sup>. As hundreds of packets are generated, the output was filtered for the protocols we were interested in (SIP and HTTP packets). Generally speaking, even the basis latency measurements were found to be higher than the latency statistics found in academic publications. There are several potential reasons for this. For example, use of dedicated high speed lines in reported trials.

The trials that invoked only local IMS services used the *UCT IMS Client* as their client application.

---

<sup>1</sup>[www.wireshark.org](http://www.wireshark.org)

The trials that invoked external Web Services used the *Firefox* web browser as their client application.

Alice's UE runs on a dual core 2.40GHz Intel CPU with a 54Mb/s 802.11g connection.  
Average IMS Registration Delay - 0.283 seconds

Bob's UE runs on a dual core 2.40GHz Intel CPU with a 100Mb/s Ethernet connection.  
Registration Delay - 0.240 seconds

Mandatory fields were marked with "\*\*"

Save Refresh Delete

Attach IFC

Select IFC... Attach

ID	IFC Name	Detach
1	default_ifc	Detach

Add SPTs to Trigger Point

Not  SIP Method PUBLISH Delete

OR

Not  SIP Method SUBSCRIBE Delete

OR

Request-URI +

AND

Not  SIP Header SIP Header Event Delete

SIP Header Content .\*presence.\*

OR

Not  SIP Header SIP Header Event Delete

SIP Header Content .\*reg.\*

OR

Request-URI +

AND

Request-URI +

Figure 7.1: Trigger Points for Presence Service

## 7.2.1 Response Times for Call-Setup

In this case we run both "Alice calls Bob" and "Bob calls Alice". No services are invoked because the filter criteria for the default profile is only set to respond to "PUBLISH" and "SUBSCRIBE" methods (as shown in figure 7.1) while "Alice calls Bob" uses only the "INVITE" method. The measurements in table 7.2.1 depict the different stages of set-up time when a call is made by sending an INVITE request between Bob and Alice. None of the calls use any IMS application servers, but the measurements in column 2 use the default service profile; this profile has the Presence Server in its Application Server list. This means that the S-CSCF adds the Presence Server to the service chain even if it is never used. In the third column a generic service profile is used. This generic profile has no application servers to invoke so the only the messages needed to set up a call are sent. *INVITE* refers to the moment at which Alice's UA sends an SIP INVITE request to Bob's address. It is equivalent to the last digit in Bob's telephone number being dialed.

*TRYING* refers to the moment at which Alice's UA receives a provisional message from

the SIP server indicating that the call setup is in progress.

*RINGING* refers to the moment at which Alice's UA first receives a SIP 180 (RINGING) response from the SIP server. IT is equivalent to when Alice first hears that Bob's phone is ringing.

Signal	Delay w/ AS(seconds)	Delay w/o AS
<i>invite-&gt;trying</i>	0.0025	0.0005
<i>invite-&gt;ringing</i>	0.7969	0.4672
<i>bye-&gt;OK</i>	0.0005	0.0006

Table 7.1: INVITE Delay: Bob as originator

No.	Time	Source	Destination	Protocol	Info
5	*REF*	128.39.21.47	193.156.19.101	SIP/SDP	Request: INVITE sip:bob@us04.nta.no, with session description
6	0.004024	193.156.19.101	128.39.21.47	SIP	Status: 100 trying -- your call is important to us
7	0.009522	193.156.19.101	128.39.21.47	SIP	Status: 101 Dialog Establishment
8	0.137424	193.156.19.101	128.39.21.47	SIP/SDP	Status: 183 Session Progress, with session description
9	0.469209	128.39.21.47	193.156.19.101	SIP/SDP	Request: PRACK sip:bob@193.156.19.74:5060, with session description
10	0.479655	193.156.19.101	128.39.21.47	SIP	Status: 200 OK
11	0.669904	128.39.21.47	193.156.19.101	SIP/SDP	Request: UPDATE sip:bob@193.156.19.74:5060, with session description
12	0.724308	193.156.19.101	128.39.21.47	SIP	Status: 180 Ringing
13	0.725187	193.156.19.101	128.39.21.47	SIP/SDP	Status: 200 OK, with session description
14	0.872945	128.39.21.47	193.156.19.101	SIP	Request: PRACK sip:bob@193.156.19.74:5060
15	0.881670	193.156.19.101	128.39.21.47	SIP	Status: 200 OK
16	9.423223	193.156.19.101	128.39.21.47	SIP	Status: 200 OK
17	9.427382	128.39.21.47	193.156.19.101	SIP	Request: ACK sip:bob@193.156.19.74:5060
18	38.674969	128.39.21.47	193.156.19.101	SIP	Request: BYE sip:bob@193.156.19.74:5060
19	38.681491	193.156.19.101	128.39.21.47	SIP	Status: 200 OK

```

User Datagram Protocol, Src Port: dsmeter_iatc (4060), Dst Port: sip (5060)
Session Initiation Protocol
  Request-Line: NOTIFY sip:alice@128.39.21.47:5060 SIP/2.0
  Message Header
    Call-ID: 1649765176
    CSeq: 22 NOTIFY
    From: <sip:bob@us04.nta.no>;tag=1025
    To: <sip:alice@us04.nta.no>;tag=587633177
    Via: SIP/2.0/UDP 193.156.19.101:4060;branch=z9hG4bK5c0b.98824963.0
    Via: SIP/2.0/UDP 193.156.19.101:8060;received=193.156.19.101;rport=8060;branch=z9hG4bK5c0b.ba33cd35.0
    Via: SIP/2.0/UDP 193.156.19.121:5064;branch=z9hG4bK2347e97526c91ff4b86532611da17872
    Max-Forwards: 15
  
```

Figure 7.2: SIP Traffic for Alice's call to Bob

## 7.2.2 Response Times for SIP Presence

In this case Alice retrieves Presence information pertaining to Bob with the use of SIP SUBSCRIBE as well as publishes her own Presence information with SIP PUBLISH. Table 7.2.2 show observed values.

METHOD	LATENCY
SUBSCRIBE	0.7482
PUBLISH	1.0582

Table 7.2: Response Time: SUBSCRIBE/PUBLISH

```

▶ Frame 1 (1181 bytes on wire, 1181 bytes captured)
▶ Ethernet II, Src: Aopen_60:10:b6 (00:01:80:60:10:b6), Dst: Vmware_91:3d:32 (00:0c:29:91:3d:32)
▶ Internet Protocol, Src: 193.156.19.101 (193.156.19.101), Dst: 193.156.19.121 (193.156.19.121)
▶ User Datagram Protocol, Src Port: 6060 (6060), Dst Port: ca-1 (5064)
▼ Session Initiation Protocol
  ▼ Request-Line: PUBLISH sip:alice@dus04.nta.no SIP/2.0
    Method: PUBLISH
    [Resent Packet: False]
  ▼ Message Header
    Route: <sip:193.156.19.121:5064;lr>, <sip:iscmark@scscf.dus04.nta.no:6060;lr;s=1;h=0;d=0;a=7369703a616c69636554064757
  ▶ Via: SIP/2.0/UDP 193.156.19.101:6060;branch=z9hG4bK2b9d.0e935f01.0
  ▶ Via: SIP/2.0/UDP 193.156.19.101:4060;branch=z9hG4bK2b9d.24589ab2.0
  ▶ Via: SIP/2.0/UDP 128.39.21.47:5060;rport=5060;branch=z9hG4bK54670974
  ▶ From: <sip:alice@dus04.nta.no>;tag=386417365
  ▶ To: <sip:alice@dus04.nta.no>
  Call-ID: 289683743
  ▶ CSeq: 20 PUBLISH
  Content-Type: application/pdf+xml
  Max-Forwards: 15
  User-Agent: UCT IMS Client
  Content-Disposition: render;handling=required
  Expires: 3600
  Event: presence
  Content-Length: 291
  P-Asserted-Identity: <sip:alice@dus04.nta.no>
  P-Charging-Vector: icid-value="P-CSCFabcd4898724700000037";icid-generated-at=193.156.19.101;orig-icid="dus04.nta.no"
  ▼ Message Body
    ▼ extensible Markup Language
      ▶ <?xml
      ▼ <presence
        xmlns="urn:ietf:params:xml:ns:pidf"
        xmlns:im="urn:ietf:params:xml:ns:pidf:im"
        entity="sip:alice@dus04.nta.no">
          ▶ <tuple
            </presence>

```

Figure 7.3: SUBSCRIBE datagram

### 7.2.3 Response Time for Web 2.0 Presence

In this case Alice imports buddies from Flickr, Facebook and del.icio.us. All principle, all imports are done with simple HTTP GET requests. Facebook responses were observed to use Javascript while Flickr and del.icio.us used more plain text and HTML in their responses. Fig 7.4 show one of the traces.

Service	Response Time
Facebook	0.5530
Flickr	0.3662
del.icio.us	0.4634

Table 7.3: Response Time for External Buddy Lists

5 *REF*	192.168.1.100	69.63.178.11	HTTP	GET /friends/ajax/friends.php?id=5309105626f1d-0&view=index&q=6nt=0&k=0&lt=0&ps=50&a=0 HTTP/1.1
18 0.553030	69.63.178.11	192.168.1.100	HTTP	HTTP/1.1 200 OK (application/x-javascript)
25 *REF*	192.168.1.100	76.13.6.175	HTTP	GET /network/social.htm HTTP/1.1
33 0.366184	76.13.6.175	192.168.1.100	HTTP	HTTP/1.1 200 OK (text/html)
39 *REF*	192.168.1.100	68.142.214.24	HTTP	GET /people/10606250@00/contacts/ HTTP/1.1
51 0.463464	68.142.214.24	192.168.1.100	HTTP	HTTP/1.1 200 OK (text/html)

Figure 7.4: Get WS Buddies

# Chapter 8

## Conclusion

This chapter presents key points learned from our study and concluding remarks based on the research done in this thesis.

### 8.0.4 IMS Latencies

The most surprising result was that basis IMS latency measurements for setting up a call between two clients attached to the same S-CSCF were an order of magnitude greater than the results published in [30]. The reason for the the discrepancy was not obvious but it is could be due to differences of implementation details on the clients. Another possibility is the overhead for session description negotiation. Figure 8.1 is from a Wireshark trace and it shows how much SDP information is sent from the UCT clients with an INVITE request. With SOAP out of the picture, the bottleneck seemed to be SIP and not HTTP. However, many of the SIP messages carry XML payloads so in a sense, the problem may have partially the same roots – XML parsing. This may continue to be a problem as long as XML is encoded as text instead of being binary encoded.

For requests targeted at services within the local IMS domain, response times for the Presence related requests SUBSCRIBE, PUBLISH and NOTIFY were 10 times faster than response times for INVITE . A significant difference in latencies was to be expected since there are less transactions involved in the presence requests.

### 8.0.5 JSLEE from a Development Environment

Early development attempts for the SSD was done with Mobicents JSLEE. JSLEE is a sound technical choice for implementing carrier-grade service orchestration. It is robust, cross-platform and has the security and low-latency features lacking in conventional enterprise environments. However, it is difficult to learn compared to servlet-based APIs like SIP Servlet. WS developers and telecom platform developers who want to develop service modules for a SLEE environment would most likely need extra training. On the other hand, perhaps only a small group of core developers need to trained to develop in a JSLEE framework. If the core developers then expose interfaces correctly then many existing third party services should be able to participate in developing a wide variety of services to be deployed in the SLEE. It is important that such

```

> From: "Bob" <sip:bob@dus04.nta.no>;tag=638132442
> To: <sip:alice@dus04.nta.no>
  Call-ID: 285176753
  CSeq: 20 INVITE
  Contact: <sip:bob@193.156.19.74:5060>
  Content-Type: application/sdp
  Allow: INVITE, ACK, CANCEL, BYE, PRACK, UPDATE, REFER, MESSAGE
  Max-Forwards: 13
  User-Agent: UCT IMS Client
  Subject: IMS Call
  Expires: 120
  P-Preferred-Service: urn:xxx:3gpp-service.ims.iccsi.mmtel
  Privacy: none
  P-Access-Network-Info: IEEE-802.11a
  Require: precondition
  Supported: 100rel
  Content-Length: 438
  P-Asserted-Identity: "Bob" <sip:bob@dus04.nta.no>
  P-Charging-Vector: icid-value="P-CSCFabcd48a4dc3200000080";icid-generated-at=193.156.19.101;orig-oi="dus04.nta.no"
  P-Called-Party-ID: <sip:alice@dus04.nta.no>
  Message Body
  Session Description Protocol
    Session Description Protocol Version (v): 0
    Owner/Creator, Session Id (o): - 0 0 IN IP4 193.156.19.74
    Session Name (s): IMS Call
    Connection Information (c): IN IP4 193.156.19.74
    Time Description, active time (t): 0 0
    Media Description, name and address (m): audio 31156 RTP/AVP 3 0 101
    Bandwidth Information (b): AS:64
    Media Attribute (a): rtpmap:3 GSM/8000
    Media Attribute (a): rtpmap:0 PCMU/8000
    Media Attribute (a): rtpmap:101 telephone-event/8000
    Media Attribute (a): fmp:101 0-11
    Media Attribute (a): curr:qos local none
    Media Attribute (a): curr:qos remote none
    Media Attribute (a): des:qos mandatory local sendrecv
    Media Attribute (a): des:qos mandatory remote sendrecv
    Media Description, name and address (m): message 24892 TCP/MSRP
    Media Attribute (a): accept-types:text/plain
    Media Attribute (a): path:msrp://193.156.19.74:24892/3dbc796c43e0;tcp
RFC 3261: Expires Header (sip.Expires), 14 bytes
Packets: 2246 Displayed: 2246 Marked: 0

```

Figure 8.1: SDP in Invite Message

services be made to adhere to a minimum level of security requirements in order to participate in blended IMS services. There could perhaps be a sandbox option where low-security web based services are allowed to interact with the user without gaining access to some sensitive services.

### 8.0.6 SIP Servlets from a Development Perspective

For the purpose of completing this study within the time constraints Mobicents JSLEE was abandoned for Sailfin SIP Servlets. The SIP Servlet API appears to offer the better development efficiency than JSLEE for the purpose of orchestrating SIP services. This holds both for dedicated IMS service developers as well as (and even more so) for third party service providers who are familiar with more mature web based technologies such as HTTP Servlets. RESTful web services without the use of SOAP is a good companion to SIP Servlets. Both follow the basic tenets of HTTP and WWW technology and may therefore make it a more viable candidate for universal adoption. The disadvantage of using SIP Servlets as the main IMS service development technology is that it does not support several different resource types as a SLEE does. Enforcement of ACID for example will have to be done as a separate module whereas in SLEE it is an integral part of the environment regardless of the protocol being used as a resource. Choosing SIP Servlets over SLEE would make it even more important to implement SLAs between IMS and third party domains. The following list summarises the most noticeable SIP Servlet advantages when implementing IMS service orchestration. Fur-

thermore Sailfin's loose coupling with the Glassfish J2EE server and NetBeans platform offers the developer many helpful tools for Web Services development as well. Here are two arguments for the use of SIP Servlets for IMS Service orchestration:

- Similarity to HTTP Servlets makes migration easier for third party web service developers to adapt
- The new *ConvergedHTTPSession* interface (new in JSR289) makes it easy to access SIP sessions from HTTP and HTTP sessions from SIP. This was important for our hybrid WS approach.

And here are two arguments against the use of SIP Servlets for IMS Service Orchestration:

- multithreading and concurrency control are important aspects of SIP sessions that are not explicitly handled by the SIP Servlet API.
- At time of writing, Sun Microsystem's Glassfish and Mobicents JSLEE container offered the only freely available SIP Servlet 1.1 reference implementations. The Sun solution was easiest to implement because it could be used independently of the SLEE. However it was only partially compatible with version 1.1 of the SIP API. The Application Router, which is the most IMS friendly addition to the SIP Servlet API is also the part of the API that was not yet implemented in Sailfin.

From the perspective of a web application developer, table 8.0.6 presents an intuitive name mapping of essential resources from the HTTP Servlet API to the SIP Servlet API for, respectively: the servlet class, the session class, the application package file and the deployment descriptor file.

PROTOCOL	SERVLET	SESSION	PACKAGE	DESCRIPTOR
HTTP	HttpServlet	HTTPSession	WAR	web.xml
SIP	SipServlet	SIPSession	SAR	sip.xml

Table 8.1: Similarities between HTTP Servlets and SIP Servlets

## 8.1 Summary

This study concludes that SIP Servlets, coupled with RESTful Web Services interactions, appear feasible for the orchestration of composite services in IMS Centered Architecture. A comparison of a basic SIP INVITE interaction (with and without the invocation of an application server), indicated that the invocation of one unnecessary a-priori service can double the response time. These results suggest that there potentially many benefits to be had by migrating from a static IMS chaining scenario to a dynamic one. If these technologies are in addition deployed within a JSLEE framework they can provide a complete orchestration solution in the IMS service layer within the required boundaries for latency and throughput but this remains to be tested. Most

importantly, this JSLEE/SIP Servlets combination can provide a more secure orchestration environment for native IMS services. In addition it provides the opportunity for safe inclusion of both existing and evolving third party services available over the web. Although the focus of this study is on third generation telecom services and web services, JSLEE would also make it possible to include legacy and proprietary services in the same server environment. So, for example, if support for CAMEL services was needed, deploying a special CAMEL server would not be necessary, one could simply attach a CAMEL RA to the SLEE container in the same manner in which the SIP RA was attached to the container for SIP services and the same way in which a HTTP RA was attached for HTTP services. The provision and maintenance of Resource Adapters could become a market niche in itself.

Our observations support Nahum's claims in [31] that a significant portion of the overhead in IMS is due to SIP signaling. The orchestration model presented in this thesis provides the following two service chaining features that are not possible in the current IMS specification:

- Services can be added to an existing SIP session.
- External, non-Parlay Web Services can be called using existing IMS data structures and interfaces.

# Chapter 9

## Further Work

The following topics are relevant to orchestration in IMS networks but were either beyond the scope of this thesis or outside of the time constraints that were given for this research. An in-depth treatment of these topics are left to further work. The areas identified for further research are:

- Statistical verification of prototype measurements, further testing and further development.
- Effective database solutions in multi-provider IMS environment.
- Security between the local IMS domain and remote service providers.

Recall that we have identified JSLEE as a promising SDP for developing services for telecom platforms. With that in mind, several of these issues could be researched within or with the help of a JSLEE framework. We will now explain each of these points in greater detail:

### 9.1 Verification of Measurements and Further Development

The measurements from this study need to be verified by way of a complete statistical treatment. Further testing should include scalability tests. The implementation itself needs to be expanded to further explore how the overhead of SIP messages can be diminished. In order to test the availability model proposed in the appended papers a long term availability study would have to be done to compile availability probabilities and identify the correct metrics.

### 9.2 Security

Several of the security challenges that present themselves in IMS service provisioning are due to inherent vulnerabilities in the SIP protocol. However in 4 we argue that security in the IMS context should not be SIP's responsibility but rather that of Diameter, HTTP and other mechanisms.

By introducing Web Services to the telecom domain we introduce additional security vulnerabilities that must also be addressed.

## 9.3 Database Solutions

One of the main challenges for IMS database solutions is the effective management of user data. As we have discussed in chapter 5, user data is central to service orchestration in IMS. At the same time, the variety of access technologies and platforms that IMS attempts to converge and the growing number of services and terminals per user make increases the complexity of the storage problem. Inevitably, there is unnecessary duplication of data. As the volume of users grow, so will the volume of services. As the notion of composite services become increasingly popular, so will the volume of services and the number of permutations with which they can all be assembled. Without effective handling of user data. This growth could have a significant negative effect on performance in the IMS network. In this light, here are some questions that need analysis.

- Is the current HSS capacity sufficiently scalable?
- How can we use the concept of distributed databases?
- Should data population be on-the-fly or a-priori?
- Which existing database solutions can we utilise?
- How can we find a balance in the trade-off between centralizing resources for effectiveness and replicating data for robustness?

The following three subsections mention three data storage technologies in use in IMS environments that should be considered in such a study

### 9.3.1 Standardisation of XDMS usage in IMS

Firstly, let us consider a database architecture that we have already discussed. Recall that XDMS is used in IMS services, like the Colibria Presence server in our implementation, but its use is not standardised in IMS. The result of this lack of approved practice is underutilization or misuse of the technology. A Presence server may for example come bundled with a XDMS. However, if a user also accesses another service from another vendor there might be no sharing of information between the two servers. Resulting in duplicated information despite the fact that the necessary technology is present.

### 9.3.2 Generic User Profile

3GPP's Generic User Profile (GUP) is meant to provide a point of convergence for the volumes of data associated to one user but distributed (and duplicated) across different network domains(for example: packet-switched, circuit-switched) and access technologies(for example: UTRAN, WLAN) [34].

### 9.3.3 Lightweight Directory Access Protocol

Lightweight Directory Access Protocol (LDAP) is used for querying directories over IP networks with the help of the Domain Name System (DNS). It is already well known in the telecom industry and given its light weight and made-for IP properties it could be a good alternative for a system that needs a standard way to discover third party SIP services.

# Bibliography

- [1] W3C Web Services Architecture Working Group. Web services architecture, working group note. Technical report, World Wide Web Consortium, February 2004.
- [2] Ivelin Ivanov. Mobicents: Jslee for the people, by the people. Internet (java.net), March 2006.
- [3] Mark Lynch. CfmX - soap vs rest benchmarks. Internet, technical blog, March 2008. <http://www.lynchconsulting.com.au/blog>.
- [4] Open IMS Core web site. <http://www.openimscore.org>.
- [5] H. Khlifi and J.-C. Gregoire. IMS application servers: Roles, requirements, and implementation technologies. *Internet Computing, IEEE*, 12(3):40–51, May 2008.
- [6] Dictionary.com. telecommunication, dictionary.com unabridged (v 1.1). Random House, Inc. (accessed: July 18, 2008).
- [7] Gartner Research. Predicts 2007: Telecom carriers will spend billions to try to survive the IP revolution. Internet, January 2007.
- [8] 3rd Generation Partnership Project. IP multimedia subsystems; stage 2 (release 8). Technical Specification Group Services and System Aspects, June 2008. 3GPP TS 23.228 v8.5.0.
- [9] Internet Engineering Taskforce (IETF). Obtaining and using globally routable user agent (ua) URIs (gruu) in the session initiation protocol (SIP). Internet Draft, October 2007.
- [10] Open Mobile Alliance. XML document management v1.1. Specification, January 2008.
- [11] Open Mobile Alliance. OMA presence simple v1.1. Specification, January 2008.
- [12] Internet Engineering Taskforce (IETF). Session initiation protocol. Request for Comments (RFC) 3261, June 2002.
- [13] Internet Engineering Taskforce (IETF). Diameter base protocol. Request for Comments (RFC) 3588, September 2003.
- [14] Michael Palmeter. Service capability interaction management (SCIM) in IMS. Internet, February 2006.

- [15] Internet Engineering Taskforce (IETF). Session initiation protocol (sip)-specific event notification. Request for Comments (RFC) 3565, June 2002.
- [16] Brenda M. Michelson. Event-driven architecture overview. Object Management Group, February 2006.
- [17] K. Mani Chandy. Event-driven applications: Costs, benefits and design approaches. In *Gartner Application Integration and Web Services Summit*, San Diego, CA, June 2006.
- [18] W3C. Web services description language (wsdl) version 2.0. W3C Recommendation, June 2007.
- [19] OASIS. Uddi specification. UDDI Spec Technical Committee Draft, October 2004.
- [20] Roy Thomas Fielding. *Architectural styles and the design of network-based software architectures*. PhD thesis, 2000. Chair-Richard N. Taylor.
- [21] 3rd Generation Partnership Project. Open service access (osa) (stage 2);(release 8). Technical Specification Group Core Network and Terminals, June 2008. 3GPP TS 23.198 V8.0.0.
- [22] Rudolf Pailer, Johannes Stadler, and Igor Miladinovic. Using parlay apis over a sip system in a distributed service platfor for carrier grade multimedia services. *Wireless Networks*, 9:353–363, 2003.
- [23] Chris Peltz. Web services orchestration and choreography. *IEEE Computer*, 36(10):46–52, October 2003.
- [24] Sun Microsystems. Jain slee 1.0 specification, January 2004. Final Release.
- [25] Tim Oreilly. Web 2.0 compact definition: Trying again. Internet, December 2006. <http://radar.oreilly.com/2006/12/web-20-compact-definition-tryi.html>, accessed 03.08.
- [26] Adel Al-Hezmi, Oliver Friedrich, Stefan Arbanowski, and Thomas Magedanz. Requirements for an ims-based quadruple play service architecture. *IEEE Network Magazine, Special Issue on Convergence of Internet and Broadcasting Systems*, 21(2):28–33, March/April 2007.
- [27] Judith Rossebø, Arlene Pearce, and Terje Jensen. On understanding availability of services based on ip multimedia subsystem. In *18th ITC Specialist Seminar, Quality of Experience*, Karlskrona, Sweden, May 2008.
- [28] 3rd Generation Partnership Project. Architecture impacts of service brokering (release 8). Technical report, June 2008. TR 23.810 v1.0.0.
- [29] Caroline Chappell. Service orchestration. Report on [lightreading.com](http://lightreading.com), August 2006. based on webinar: Supporting New Service Delivery Models Using ‘Service Orchestration’.

- [30] Dragos Vingarzan and Peter Weik. End-to-end performance of the ip multimedia subsystem over various wireless networks. In *Wireless Communications and Networking Conference*, pages 183–188. IEEE, 2006.
- [31] Erich Nahum, John Tracey, and Charles P. Wright. Evaluating sip proxy server performance. In *17th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Urbana-Champaign, Illinois, June 2007.
- [32] Dan Davis and Manish Parashar. Latency performance of soap implementations. In *2nd IEEE International Symposium on Cluster Computing and the Grid (CCGRID)*, pages 407–412, Berlin, Germany, May 2002.
- [33] 3rd Generation Partnership Project. Ip multimedia (im) subsystem cx and dx interfaces (release 8). Technical Specification Group Core Network and Terminals, June 2008. 3GPP TS 29.228 V8.2.0.
- [34] 3rd Generation Partnership Project. 3gpp generic user profile (gup) architecture (stage 2);(release 7). Technical Specification Group Services and System Aspects, June 2007. 3GPP TS 23.240 V7.0.0.

# Appendix A – List of Acronyms

ACID	Atomicity, Consistency, Isolation, Durability
AJAX	Asynchronous JavaScript and XML
AKA	Authentication and Key Agreement
AR	Application Router
AS	Application Server
AVP	Attribute Value Pair
BPEL	Business Process Execution Language
CAMEL	Customised Applications for Mobile networks Enhanced Logic
CFC	Cold Fusion Components
CFM	Cold Fusion Markup Language (also called CFML)
CNF	Conjunctive Normal Form
CSCF	Call Session Control Function
DCOM	Distributed Component Object Model
DNF	Disjunctive normal form
EDF	Event Driven Architecture
GPL v2/v3	GNU General Public License version 2/3)
GRUU	Globally Routable UA URI
GUP	Generic User Profile
I-CSCF	Interrogating Call Session Control Function
iFC	Initial Filter Criteria
IMPU	IP Multimedia Private Identity
IMPU	IP Multimedia Public Identity
IMS	IP Multimedia Subsystem
IM-SSF	IP Multimedia Services Switching Function
ISC	IMS Service Control
HLR	Home Location Register
HSS	Home Subscriber Server
JAIN	Java API for Intelligent Networks
JSLEE	JAIN Service Logic Execution Environment
MTU	Maximum Transmission Unit
MSISDN	Mobile Station ISDN
OASIS	Organization for the Advancement of Structured Information Standards
OMA	Open Mobile Alliance

OMG	Object Management Group
OSA	Open Services Architecture
OSA-SCS	OSA Service Capability Server
OSS	Operational Support System
P-CSCF	Proxy Call Session Control Function
PSI	Public Service Identities
QoS	Quality of Service
RPC	Remote Procedure Call
RSS	Really Simple Syndication
RTT	Round Trip Time
SAML	Security Assertion Markup Language
S-CSCF	Serving Call Session Control Function
SB	Service Broker
SCIM	Service Capability Interaction Manager
SDP	Service Delivery Platform
SDP	Session Description Protocol
SLA	Service Level Agreement
SLF	Subscription Location Function
SIMPLE	Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol (SOAP 1.1 only, SOAP 1.2 is not an acronym)
TAPI	Telephony Application Programming Interface
UA	User Agent
UDDI	Universal Description, Discovery and Integration
UE	User Entity
UMTS	Universal Mobile Telecommunications System
UPSF	User Profile Server Function (HSS for fixed networks)
URI	Uniform Resource Identifier
UTC	University of Cape Town
UTRAN	UMTS Terrestrial Radio Access Network
XCAP	XML Configuration Access Protocol
XML	Extensible Markup Language
XDMS	XML Data Management Server
WS	Web Service(s)

## Appendix B – ITCss Paper

This appendix contains the reviewed academic paper *On Understanding Availability of Services Based on IP Multimedia Subsystem* by Judith Rossebø, Arlene Pearce and Terje Jensen. This paper was presented by myself at the 18th International Teletraffic Congress Specialist Seminar (ITCss) on Quality of Experience in Karlskrona Sweden, May 29, 2008 and is to appear in the conference proceedings.

# On Understanding Availability of Services Based on IP Multimedia Subsystem

Judith E. Y. Rossebø, Arlene Pearce, and Terje Jensen

**Abstract**—Availability has commonly been considered as an atomic property, indicating the average amount of time a system is working as planned. This, however, should be further detailed considering that more compound services are being deployed. Expectedly, such situations are met when federating services based on the IP Multimedia Subsystem. Then, certain parts of the service may work to the complete satisfaction of the user, while other parts are not quite up to the planned behaviour. It is usually also included as the planned behaviour that unauthorized users should not get access to the services or relevant data. Hence, two essential characteristics of availability are accessibility and exclusivity. This paper presents and discusses a conceptual model for service availability intended to capture characteristics of these service types and demonstrates how the model can be used to analyze compound/composite services. Single provider and multi-provider configurations are exemplified.

**Index Terms**—Availability concept, federated services, IP Multimedia Subsystem, Quality of Experience.

## I. INTRODUCTION

**A**VAILABILITY is a central characteristic in service delivery. In fact, in today's society available telecom services are pivotal for several enterprises, e.g., banking and financial services. The concept of service availability, however, goes beyond telecom. Often, consequences are serious if even parts of telecom or computer systems are unavailable when their services are needed.

Motivated by cost savings, more and more telecom services are being migrated from deployment over dedicated networks to deployment over a common IP-based infrastructure. There is a need to ensure that the IP-based infrastructure can support services with acceptable availability characteristics.

Traditionally, the notion of availability has been defined as the probability that a system is working at time  $t$ , and the availability metric has been given by the uptime ratio, representing the percentage of time that a system is up during its lifetime [1]. Accompanying this interpretation, failure reporting procedures have also been described, e.g. [2] for Public Switched Telephony Network, PSTN. This understanding has served well for describing and analyzing availability

J. E. Y. Rossebø is with Telenor Research and Innovation, NO-1331 Fornebu, Norway. She is also affiliated with The Norwegian University of Science and Technology (NTNU) (e-mail: Judith.rossebo@telenor.com).

A. Pearce is with Telenor Research and Innovation, NO-1331 Fornebu, Norway. She is also pursuing MSc degree at the University of Oslo (e-mail: pearce@pvv.ntnu.no).

T. Jensen is with Telenor Research and Innovation, NO-1331 Fornebu, Norway, and the Norwegian University of Science and Technology (NTNU) (e-mail: terje.jensen1@telenor.com).

This work has partially been funded by the Research council of Norway project SARDAS (152952/431) and partially within the Eureka project Mobile Fixed Convergence in Multiaccess Environment, Mobicome.

of services delivered in dedicated networks such as for voice services in the PSTN/ISDN. However, for describing service availability characteristics and analyzing availability of services in the vastly distributed environment in which IP-based services are deployed, an enhanced notion of availability is required.

Considering the emerging range of IP-based services being delivered in public and private networks today, several challenges follow from the traditional understanding of availability. This paper addresses two challenges. The first challenge is that even with a high mean rate of availability, failure that occurs during peak service request periods will result in high operational loss. One such scenario is a web service with 99.999% average availability that loses connectivity for 5 minutes during peak sales of concert tickets. Such bursty behaviour patterns could be seen for several of the services [3]. The second challenge is that when presented with a set of service components, a user may have different expectations of quality for each component. One example is a buddy list with presence information fed by an IP Multimedia Subsystem, IMS. Different categories of buddies may be defined, say work-related and leisure-related. A user may tolerate lower quality weather forecast or tv-guide services she typically uses in leisure-mode while she may expect near perfection from a stock-ticker service used in work-mode. Similarly, reliable presence information pertaining to her colleagues/employees may be more important to her than reliable presence information pertaining to leisure-related contacts.

In the multi-application environment implied by IMS, several features may contribute to the overall user experience. For example, the user interface may collect parts of presence information, location-dependent data, calendar tasks, and other service components. Different parts of the user interface may be updated by different servers. Hence, the user experience is collated from different sources. Moreover, the different parts of the user interface may have different weights in the experience depending on the user tasks.

We focus on the problem of considering the variability of the contributions of the different parts involved in the services to the overall availability of a service. Issues are the adequacy of the current availability concept, the definition of availability and the availability management. Quality of Experience, QoE, should then be related to these issues as addressed by this paper.

This paper addresses these issues. The service availability concept model motivated and introduced in [4] is presented and exemplified by a set of cases. As services grow in com-

plexity [5], further aspects of availability need to be covered. This paper presents and elaborates on a conceptual model for service availability providing a case study to demonstrate the applicability of the model to service provisioning in a distributed IMS service environment.

Sect. II provides a brief introduction to the enhanced service availability concept. An overview of IMS follows in Sect. III. Sect. IV exemplifies how the enhanced service availability concept can be applied for a federated presence implemented on IMS.

## II. ENHANCED SERVICE AVAILABILITY CONCEPT

### A. General Motivation

The setting for the enhanced service availability concept is derived from the fields of dependability and security. As explained in [6], availability has been treated by the field of dependability and the field of security with different definitions and understandings of what availability is [7], [8], [9], [10].

The definition of availability used as a basis for the enhanced service availability concept is: The property of being accessible and usable on demand by an authorized entity [8], [10]. This definition captures the integral part of securing availability by ensuring access to authorised users while also addressing the aspect of a service being usable in addition to the traditional aspect of readiness for correct service.

The notion of service availability has been further refined using this definition as a basis, to include addressing the *exclusivity* aspect of ensuring that a service is provided to the authorized users *only* [4]. This aspect is important because a system must know how many users are expected to access a service at a given time as well as how long the users are expected to access the service. The of users accessing at a given time and the session durations can be used to calculate the penetration and usage values. These values could be applied when dimensioning and as basis for ensured performance levels. If the means to ensure that authorized users *only* are accessing a service is too weak, and unauthorized users are able to access a service, the service availability for authorized users may be affected.

As established in [7], availability is affected by means and threats. The conceptual model of dependability consists of three parts: the attributes of, the threats to and the means by which dependability is attained [11] and provides a basis for the service availability conceptual model as motivated in [6]. In order to classify threats to availability and means to achieve availability in a security setting, we are also motivated by the approach used in the security field of risk analysis and risk management as in [12], [13].

This is because incidents resulting in loss of availability do not necessarily escalate into faults and therefore classification of means in terms of faults may become insufficient for availability analysis. An example is the hijacking of user sessions by an attacker or group of attackers, preventing the authorised user or group of users from accessing the service. This incident results in loss of service availability for a set of users, without incurring a fault in the system. An

unwanted incident is defined in [14] as an incident such as loss of confidentiality, integrity and/or availability. A fault is an example of an unwanted incident. The service availability conceptual model therefore classifies the means to achieve availability in terms of countering unwanted incidents.

In [15], the threats to dependability are defined as faults, errors and failures, and these are seen as a causal chain of threats to dependability:

fault  $\rightarrow$  error  $\rightarrow$  failure

This understanding of threats serves nicely in the dependability model, however, as service availability may be reduced e.g. by a denial of service attack without incurring a fault, error or failure, we apply the definition of threat, as defined in [10]: a threat is a potential cause of an unwanted event, which may result in harm to a system or organisation and its assets.

Services can exist in numerous degraded but operational/-usable/functional states between up and down or correct and incorrect. For example, an online newspaper may behave erratically with slow response times for displaying articles browsed without going down or becoming completely unavailable. This means that a more fine grained measure of availability is needed than pure up or down.

It should be possible to describe various states of availability in order to specify the extent of which a reduction of service quality may be tolerated. The service availability metric should take into account, for example, measurement of different levels of degradation of services in order to analyze more closely how well user requirements are fulfilled, as well measuring the ability to adequately provision a service to all of the authorised users requiring the service at a given moment. Such a metric should take into account the appropriate set of parameters, not just the usual average based on the mean time to failure (MTTF) and the mean time to repair (MTTR). In section IV below, we provide a set of parameters for measuring the availability of an IMS presence service.

### B. Enhanced Basic Notion

The enhanced notion of service availability encompasses both exclusivity, the property of being able to ensure access to authorised users only, and accessibility, the property of being at hand and useable when needed. Exclusivity involves ensuring that unauthorised users cannot interrupt, hijack, or prevent the authorised users from accessing a service. The focus is on preventing the denial of legitimate access to systems and services by prohibiting unauthorised users from interrupting, or preventing authorised users from accessing services. The aim is to ensure access to users while keeping unauthorised users out. Some of the means to achieve exclusivity address ensuring access for authorised users and others address techniques for preventing unauthorised users from accessing or interrupting services, e.g. by monitoring to discover unwanted traffic and blocking this traffic from unauthorised users.

Accessibility is defined as the quality of being at hand and usable when needed. We divide accessibility properties into three major areas: timeliness, correctness and usability.

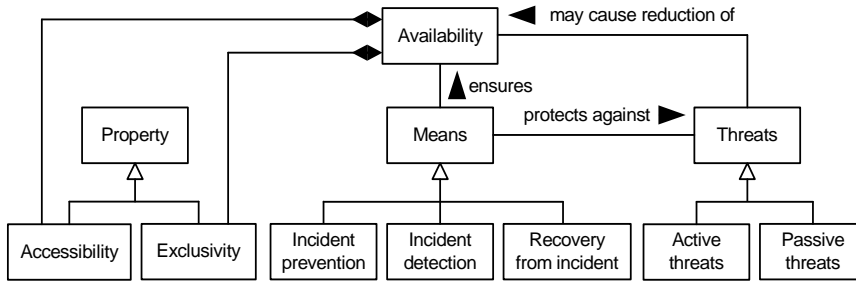


Fig. 1. Conceptual model for service availability

Timeliness is the ability of a service to perform its required functions and provide its required responses within specified time limits. Usability is concerned with the users perception of the service, and the ease of use of the service. The measure of correctness of a service may differ widely between different kinds of services.

Consider an online payment service. From the viewpoint of a user at a given point in time, we could say that the quality of the service is either 1 or 0 depending on whether the user gets a useful reply (e.g. confirmation) or not (e.g. timeout). (Over time this can be aggregated to percentages expressing how often one of the two kinds of responses will be given.)

These considerations motivate a notion of service degradation [16]. Service degradation can be defined as reduction of service accessibility. Analogous to accessibility, we divide service degradation into timeliness, usability and correctness degradation. These are mutually dependent on each other. For example, graceful degradation in timeliness may be a way of avoiding correctness degradation if resources are limited, or the other way around.

In summary, the overall conceptual model can be depicted as in Fig. 1 (illustrated in UML 2.x format [17]). Availability is affected by means and threats. Means can ensure availability by protecting against threats. Threats may lead to unwanted incidents which may cause reduction of availability.

By means to ensure availability we address protection of the service from incidents leading to a loss of availability. We have categorized the means into i) incident prevention: how to prevent incidents causing loss of availability (e.g. access control, integrity protection ensuring graceful degradation); ii) incident detection: how to detect incidents leading to loss of availability (e.g. traffic inspection, audit logs); and, iii) recovery from incident: the means to recover after an incident has lead to a loss of availability (e.g. system adaptability, robustness, maintainability, redundancy).

Threats may originate on the inside (inside attackers) or the outside (outside attackers) of the system. The impact of threats varies with the nature of the threats; some threats may result in degradation of the service, others in complete loss of service. For the full motivation and explanation of the model, see [6].

### C. Decomposing Availability

Based on the conceptual model, the availability of a service can be analyzed with respect to exclusivity and accessibility aspects. On an abstract level, a mathematical representation can be given as follows; Let  $A$  denote a service with an availability property for a user group  $U$ , and let  $X$  denote the availability metric for service  $A$ . We represent  $X = (x_1, \dots, x_n)$  as an  $n$ -tuple where  $x_i$  is a measure of an aspect of availability. These include behavioural, preventive and correctness aspects. By this we mean that  $x_i$  describes requirements for a particular availability aspect. The minimum requirement for each  $x_i$  must be satisfied in order to fulfil the total availability requirement  $X$ . Using the conceptual model this idea can be refined as follows: We represent  $X$  as a tuple  $X = (X_1, X_2)$  where  $X_1$  measures the exclusivity properties, and  $X_2$  measures the accessibility properties. Essentially, the aim is to describe the degree of accessibility and exclusivity that is sufficient for the user to be able to activate and use the service. The purpose of service availability metrics is to measure how well service availability requirements have been met.

For example, exclusivity metrics could measure how well the following requirements are met:

- The probability that an authorised user is denied access to the service at a given time  $t$  should be less than  $x$ .
- The probability that an unauthorised user obtains access to the service at a given time  $t$  should be less than  $y$ .
- User  $u$  should be prohibited from accessing service  $s$  when user  $v$  is using the service.
- The of intrusions at a given time  $t$  (e.g. during a critical moment) should be less than  $z$ .

Based on these requirements, we have the following measures of aspects of exclusivity:

- The probability that an authorised user is denied access to the service at a given time  $t$ .
- The probability that an unauthorised user obtains access to the service at a given time  $t$ .
- The probability that unauthorised user  $u$  obtains access to service  $s$  when user  $v$  is using the service.
- The of intrusions at a given time  $t$ .

Similar requirements may be defined for accessibility.

### III. IP MULTIMEDIA SUBSYSTEM (IMS)

IMS has been promoted by several international bodies as a future platform for providing services. It is access agnostic in the sense that services could be provided over any access type and to any device. That is, as long as the device is capable of supporting the proper client behaviour.

#### A. Layered Architecture

A layered architecture has been applied for defining IMS, see Fig. 2. In the core part, we find common session control and common user data. In IMS terms these are referred to as Call Session Control Function (CSCF) and Home Subscriber Server (HSS), respectively. There are several types of CSCF supporting roaming users, interconnecting between domains and emergency sessions, although these are not depicted in Fig. 2.

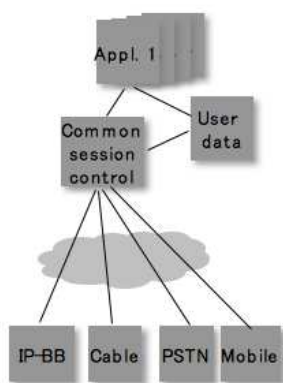


Fig. 2. Layered architecture of IMS

HSS stores user identities and user profiles. Both private and public identities are defined and a user/individual can have a set of user profiles. A profile explains which applications are to be invoked and how services are to be executed.

A range of applications can reside in the common IMS core. Examples of application types are group list managers, Centrex, location, handover support (WLAN 2G/3G).

Main protocols are SIP-based and DIAMETER-based; The former for service/session control and the latter for data access and charging. IMS has, however, defined additional parameters and attributes compared with the original IETF SIP.

The client side is not directly illustrated in Fig. 2. For IP-based terminals, an IMS client would be implemented in the terminal. Then, there will be a session running between the terminal and the CSCF. For traditional circuit-switched networks, such as PSTN and GSM-CS, the client could be said to be implemented in the switching control. That is, the media gateway controller would run session control with the CSCF. In this manner, an IMS installation could be considered to fill similar roles as Intelligent Networks do today.

As shown in Fig. 2, the effect is that multiple applications can be accessed and used through different access types. Or, in other words, it allows a seamless experience over different access and terminal types.

There are several ways in which the key components SIP Application Server, User Agent (UA), CSCF and Home Subscriber Server (HSS) can communicate with each other. This provides flexibility when designing architecture for service creation and orchestration. Fig. 3 shows existing interfaces that can be utilised in designing a new architecture.

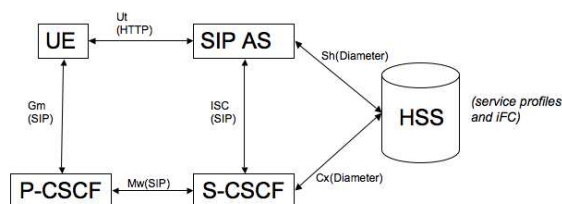


Fig. 3. Interfaces involved in service composition

The SIP AS is an integral part of the IMS specification. Note that it has direct access to the users profiles in the HSS. User profiles can in turn have service profiles. The SIP AS can also take several different roles in the SIP session initiation chain and these roles can be leveraged in service orchestration.

As a set of applications may be invoked for a session, there could be a need for allowing these applications to inter-play. For example, the Centrex application may want to know the location of a user to decide upon further service execution. One example is that different rules should be followed if the user is in the office area compared to when the user is in the car. For this purpose, a Service Capability Interaction Manager has been identified. However, it has not been described in detail, allowing for several implementation options. In particular, there are discussions on whether the Service capability Interaction Manager should be implemented in a centralized, distributed or hybrid manner.

#### B. Presence Service

One application that we analyze with respect to the service availability concept model presented above, is the presence service. Basically, the presence server collects information about a set of users and presents this information to pre-defined users. Schematically, it can be depicted as in Fig. 5. The users from whom presence information is requested are called Presentities. Presence sources are defined as nodes reporting the presence information. Examples of presence sources are clients in handsets, mail/calendar servers, network elements and indications given through web portals.

This information is collected by the presence server and reported to the pre-defined watchers. Again, different watcher types can be defined, such as clients on handsets and other application servers.

As also shown in Fig. 5, not only users can be reported through this mechanism, but also other types of items, such as stock prices that a user has subscribed to monitor. There are also providers utilising the same mechanism for presenting advertisements, reminders or other offers.

As a result, the list of items provided by the presence server can be grouped into sets that have different expectations as seen by a user. These may also vary during the day, for example as some items may be more related to work, while others are more related to leisure, family or social groups.

The different presence items may be updated from different sources. This means that several different service providers as well as others may be involved. For example, a user may set up buddies that are subscribers of other service providers. This implies that the different providers must interact, and cooperate. An immediate case is to incorporate Facebook friends into an IMS based buddy list as illustrated in Fig. 4.

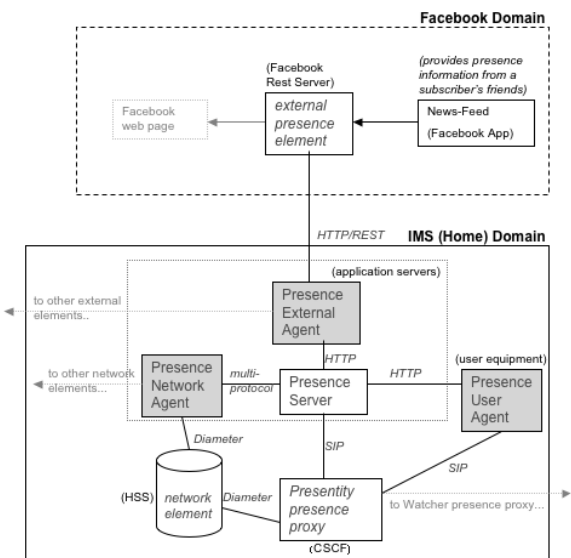


Fig. 4. Facebook as External Presence Element

Presence information, as defined by [18], conveys the ability and willingness of a user to communicate across a set of devices. A presence service is a system that accepts, stores and distributes presence information to interested parties, called watchers. A presence protocol is a protocol for providing a presence services over any network.

The presence server is located in the home network of the user that the presence information is relating to (called presentity). It would commonly include both logic and data storage. Key capabilities are collecting, composing and filtering presence information. Filtering could be used when deciding which presence parameters should be presented to an actual watcher application. This could also be used when only the parameters that have changed since the previous update should be forwarded. Filters can define which tuples are watched

(e.g. all that has contact address equal to tel:user@domain), attributes to be forwarded to a watcher and triggers when notifications should be sent.

Watcher information shall be collected by the presence server, which allows a presentity to obtain that information. Any presence information that the presence service is not able to interpret shall be handled in a transparent manner.

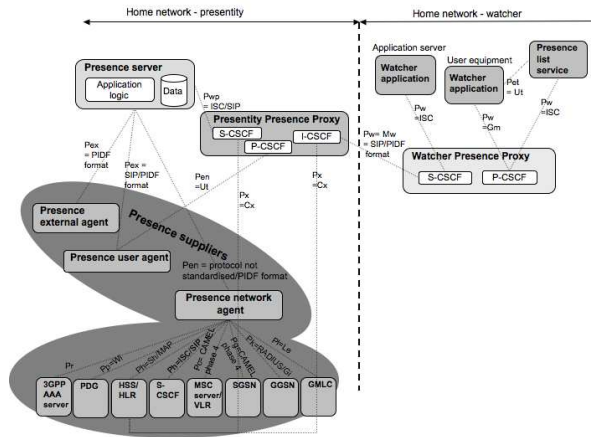


Fig. 5. Functional components for presence (adapted from 3GPP TS 23.141 [19]).

Subscription authorization policy shall be provided by the presence server. This tells which watchers that are allowed to subscribe to which of a presentitys presence information.

A number of suppliers of presence information can be relevant:

- Presence network agent: providing presence information from network elements. A range of network elements and corresponding protocols are shown in Fig. 5. Examples of presence information obtained are i) from GGSN PDP context activation, de-activation, ii) from SGSN attached, not reachable for paging, detached, routing area update, iii) from MSC server attach, detach, location area update, call setup, idle, connected, busy, etc., iv) from 3GPP AAA server WLAN UE attaching/detaching, tunnel establishment/removal.
- Presence user agent: providing presence information on behalf of a principal. This may be located in the terminal or in the network (e.g. when the terminal does not have a proper user client installed). In case it is located in the network, it must be within the presentitys home network.
- Presence external agent: providing presence information by elements outside the providers network. This takes care of any interworking and security issues and resolves location of the presence server. Examples of information supplied by the presence external agent include i) third party services (calendar applications, corporate systems, etc.), ii) internet presence services, iii) other presence services.

The presence proxy can be used to locate the presence server (will make a lookup in the HSS) that is to be used for a given user/presentity.

The watcher entities are divided into watcher presence proxy and watcher applications. The watcher presence proxy carries out functions such as authentication of watchers.

Watcher applications can be located in i) user equipment, ii) application server, and, iii) external to providers domain. The same interface is used both for requesting monitoring and for fetching information. In both cases all or a subset of a presentity's information can be transferred (e.g. referring to a given filter, only parameters changed since last notification, etc.).

A presence list server (an SIP server) can also be involved keeping information regarding grouped lists of watched presentities. This enables a watcher application to subscribe to presence of a group of users/presentities by a single SUBSCRIBE transaction. A presence list server also stores and manages filters associated to presentities in the presence list. Filter shall be attached to individual SUBSCRIBE transactions.

A watcher application sends a SIP SUBSCRIBE to Event:presence addressed to the presentity's SIP URL to subscribe or fetch presentity's presence information. This SUBSCRIBE request will be handled by the IMS core elements reaching the presence server. The presence document is provided by the presence server to the watcher application using SIP NOTIFY (see Fig. 6). The SIP NOTIFY may be triggered by a change of the presentity's status, notified by any of the presence suppliers via the corresponding interfaces and message types.

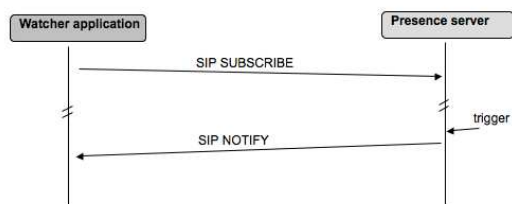


Fig. 6. Schematic message flow between a watcher subscribing to updates of presence information

Both of the accessibility and the exclusivity aspects of service availability discussed in Sect. II are related to the presence item list. For each of the different groups in the item list, the degree of exclusivity achieved and accessibility achieved may differ. For example, during leisure time, buddies related to work may not require to receive updated information on a user's whereabouts. On the other hand, during office hours, it may be considered crucial by the user to receive updated information on other users (buddies) whereabouts.

Level of detail may also differ for the different items. For example, for certain buddies, the user is allowed to see their locations, while for others the locations must not be displayed.

Again, these rights may vary over time.

Hence, the user's desktop can be looked upon as consisting of a set of fields. For each of the fields the user has a certain expectation of correct behaviour, both in terms of that the fields content should be correct and that it should not be revealed to other non-qualified parties. On a conceptual level, each of the items is expected to be up to date with respect to specific timeliness requirements. The overall user's experience might then be divided into expectations for each of the fields. This leads to decomposition into fields with separate availability requirements.

A time stamp field in the message format may be used as a rudimentary security mechanism to prevent replay attacks e.g. launched for the purpose of capturing user credentials for unauthorized access to a service. For example, tuples whose time stamp are older than the time stamp of the most recently received presence document should be discarded.

Attributes shall be mapped to separate tuples that have unique identifiers. In case different attribute values should be shown to different watchers, a set of tuples must be created that contain the same attribute. The subscription authorisation policies give the association of tuples to different watcher groups, that is, which watchers can access which presentity information.

Subscription authorisation lists can be divided into the following categories:

- 1) blocking; watchers not allowed to access any presence information related to the presentity.
- 2) personal; explicitly identifying watchers
- 3) general; groups of watchers who are not necessarily known by the presentity, e.g. all watchers

The list categories are evaluated in the order

1) → 2) → 3)

#### IV. CASE – AVAILABILITY ASPECTS OF THE ENHANCED PRESENCE SERVICE

The enhanced presence service based on IMS is assumed to combine a set of different features, such as location information as well as advertisements, see Fig. 7. There is also support to include buddies that are subscribers of other domains, e.g., other service providers.

##### A. Single-Provider Case

The first case is depicted in Fig. 8. Here, it is assumed that all users are within the same service provider domain. It is also assumed that group lists are stored by the presence server.

In a straight-forward manner, for the overall service to work, all components have to be in operation. For the configuration shown in Fig. 8, user 2 and user 3 are attached to the same network element. Hence, it suffices that this element is in operation. A block diagram could then be set up in order to analyse the service availability with respect to accessibility.

Suppose, however, that not all users are of equal importance for user  $N$ . Then, accessibility estimates can be made for each user (i.e. items) on the presence list, e.g.  $A_i$  for user  $i$ . Average

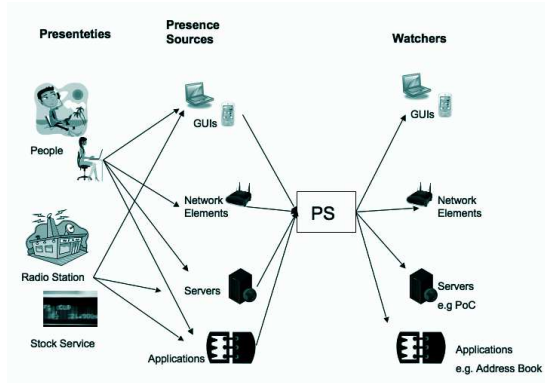


Fig. 7. Presence server conveying information about users to defined watchers

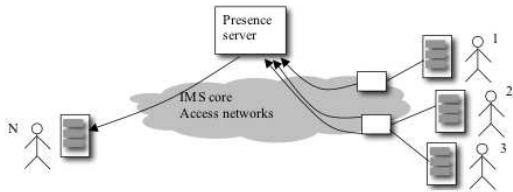


Fig. 8. Case I: user with 3 buddies for presence listing

accessibility is then found as  $\sum_i (A_i \cdot \alpha_i) / \sum_i \alpha_i$ , where  $\alpha_i$ , is the level of importance associated with user  $i$ .

### B. Multi-Provider Case

Providing different levels when all users are within the same provider domain seems contrived. When the different users are in different domains, different levels may become more relevant. An example is depicted in Fig. 9.

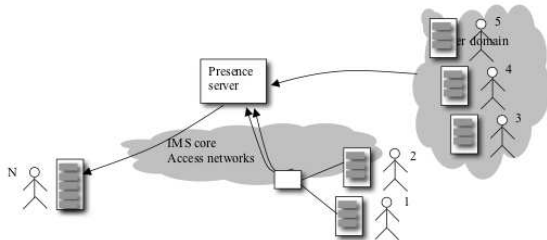


Fig. 9. Case II: user with buddies in different provider domains

Again, a block diagram could be put up for each of the items in the list. For this configuration, however, certain details regarding implementation of presence services in other domains would likely not be available. This implies that accessibility levels would then be part of the Service Level Agreements (SLAs) established between providers. It might also be more suitable to state different levels towards user  $N$

for the different user groups in this configuration. However, this will probably depend on the SLA terms. Similar expression as above follows, although an index  $j$  indicates the group of buddies:  $A_j = \sum_i (A_{ij} \cdot \alpha_{ij}) / \sum_i \alpha_{ij}$

In some cases, there may not be any pre-established SLA between the providers. It then becomes a business risk evaluation whether a service provider wants to state any performance levels in the service description to user  $N$ . Potentially, there may be differentiated levels for the different groups,  $j$ .

Level of importance related to a user,  $\alpha_{ij}$ , may vary depending on the role of the user,  $N$ . For example, during working hours, it is more important to follow work colleagues, for example, involved in the same project, than outside working hours. As projects come and go, the colleagues involved will also vary, requiring that this information is easily updated frequently.

### C. Parameters Included in Enhanced Presence

The parameters given for a presence item may include nickname, mode, location, as well as others. Typically, location could be given with different level of granularity. In some cases, e.g. during roaming, the location may also be unknown. On the other hand, there are certain usages where location is considered as very important, such as following children in kindergarten and following some mental patients.

For user  $N$ , different levels of importance could then be attached to the different parameters. These importance weights may also differ for the different presence items as shown in Fig. 10.

As these parameters may be pushed or pulled from different sources, different response or delivery times would result. In some respects, this is similar to the design of a web page consisting of a set of objects. In order to improve QoS and network performance, presence parameters should be delivered in appropriate sub-groups. That is, waiting for the last presence parameters before sending an update to the users buddy list (watcher) would likely result in too long response times and degraded QoE.

This sub-grouping is particularly a critical aspect when information is collected from vastly different sources, some residing in semi-real-time environments while others within best-effort environment. In effect, this balances the different aspects of availability (correctness, timeliness and usability) as described in Sect. II.

### D. Use of Presence Items for Other Purposes

Having configured means for controlling presence items on a terminal, one could utilise this for other purposes as well, such as advertising and time-related special offers. One such implementation has been tested in a real user environment in Finland, in the SmartRotuuri project [20]. The service implemented included highly personalized direct marketing to customers mobile phones.

When users accept such commercial activities, a service provider would then likely have an agreement with a of companies for delivering the advertisements/offers. Then, there

will also be requirements on accessibility for providing this feature. Again, block diagrams could be made assisting the accessibility evaluation.

### E. Exclusivity

From the outset, overall exclusivity could be analysed in a similar manner as for accessibility. That is, either looking at the overall average or looking at each presence item individually. However, the analysis approaches would likely differ as there are often other types of threats affecting the exclusivity aspect.

The aim with respect to the exclusivity aspect is to ensure that authorised users only should have access to presence items that they subscribe to. Allowing unauthorised users to access presence items may have a negative effect on accessibility, but also, may be in violation of privacy directives in the jurisdiction.

Commonly, for the presence service there will be stricter requirements regarding exclusivity than for accessibility. This is mainly due to privacy aspects, avoiding any third party being able to follow a users actions. However, it is also important to ensure that the authorised users are not prevented or interrupted from accessing the presence items. Activity initiated by unauthorised users can adversely affect the accessibility aspects.

It may also be important to measure whether a rogue service provider is interfering with the availability of presence items delivered from another service provider.

It would also be a business decision on whether to provide exclusivity measures for the average or for groups of items. Some statistical aggregation measures could also be defined. How the exclusivity aspects are measured may also depend on the means that are deployed for ensuring exclusivity.

### F. Threats and Corresponding Means

Considering the different servers, network elements, protocols, data, etc. that are involved, various threats would be relevant. Examples of threat agents are malicious users, rogue service providers, unauthorised users masquerading as authorised users to obtain information on another users whereabouts. The vulnerability of the presence service to distributed denial of service attacks should be evaluated.

Hence, combinations of means to address the threats, covering protection, detection and recovery will be required. Preventative means will involve access control measures, for ensuring that authorised users only have access to the items. Monitoring of activity to detect malicious user behaviour could also be deployed. Essentially, preventative mechanisms will try to eliminate the possibility of attacks by threat agents or to enable the presence service nodes to be able to endure attacks without denying service access to authorised users. Detection and recovery will involve detecting attacks on nodes of the service, or against specific users, and responding immediately to restrict impact.

ETSI TISPAN<sup>1</sup> has developed a threat vulnerability and risk assessment (eTVRA) method and tool that may be used to analyse the presence service [22]. Using the eTVRA method and tool, the threats to availability of the presence service can be analyzed and a set of recommended countermeasures can be identified that when implemented will reduce the overall risk.

### G. User Experience

As mentioned above, expectations from users may vary for the different presence items and the different parameters for each item. Moreover, these weights may vary over time. Schematically, the actual user experience could be estimated by multiplying the weights and the actual obtained accessibility and exclusivity measures as depicted in Fig. 10.

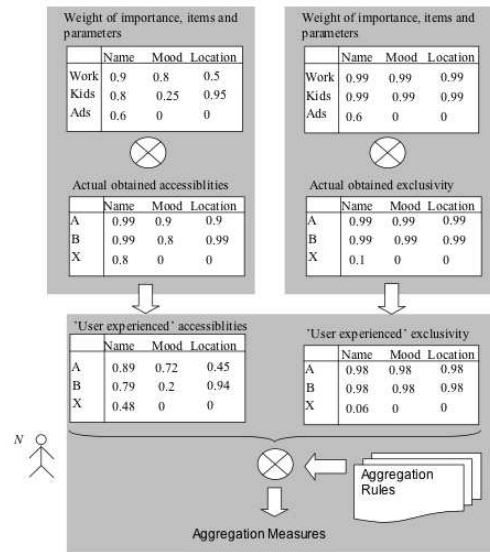


Fig. 10. Schematic procedure for estimating actual user experience (for illustration only)

Rules for aggregating experience parameters would likely be adapted to different purposes, such as statistical performance levels stated in SLAs, for Key Performance Indicators, and so forth.

It still remains to figure out how aggregated measures should be calculated and how many measures should be defined. Theoretically, based on questionnaires, a single parameters could suffice, say providing an estimate for the question, "On a scale of 1 – 0, how satisfied are you with the presence service?"

<sup>1</sup>The European Telecommunications Standards Institute (ETSI) is an independent, non-profit organization, with a mandate from the European Commission to undertake standardisation of Information and Communication Technologies (ICT) within Europe alongside its partners CEN/ISSS (European Committee for Standardization/Information Society Standardization System) and CENELEC (European Committee for Electrotechnical Standardization). ETSI TISPAN [21] is taking a leading role in developing standards for fixed mobile convergence (FMC).

Additional requests for following the performance levels are also needed. One example is monitoring key quality indicators related to selected aspects of the business. Another example is accessing performance levels when corresponding parameters are specified in the SLA between actors.

## V. CONCLUSION

This paper elaborates and exemplifies a conceptual model for availability. A key point is to include both the accessibility and the exclusivity aspects of the service availability measure. Hence, only the authorized users should be ensured access to the service, and with the proper service levels. So far, it seems that exclusivity is an aspect of availability that has rarely been included in the literature. However, the concept presented here shows where exclusivity fits in with an enhanced notion of service availability.

The enhanced notion seems even more important when considering collaboration between providers, and also between different roles of the same user. Proper service composition, also referred to as orchestration, then becomes even more important and challenging.

A schematic example is provided through the enhanced presence service realised by IMS. Considering the federated nature of the presence service, a range of challenging aspects need to be addressed, including differentiation of presence items and parameters for an item while also handling multiple sources of presence data.

Among items for potential further work is the task of conducting user experiments to estimate QoE related to different items of the presence service. Other items are to assess weights and further details for the schematic procedure outlined in Fig. 10.

## REFERENCES

- [1] S. M. Ross, *Introduction to probability models*, 6th ed. Academic Press, 1997.
- [2] P. Enriquez, A. B. Brown, and D. A. Patterson, "Lessons from the PSTN for dependable computing," Workshop on Self-Healing, Adaptive and self-MANaged Systems (SHAMAN), 2002.
- [3] D. Clark, W. Lehr, and I. Liu, "Provisioning for bursty Internet traffic: Implications for industry and Internet structure," MIT ITC Workshop on Internet Quality of Service, 1999.
- [4] J. E. Y. Rossebø, M. S. Lund, K. E. Husa, and A. Refsdal, "A conceptual model for service availability," *Quality of Protection: Security Measurements and Metrics*, vol. 23, 2006.
- [5] W. A. Arbaugh, W. L. Fithen, and J. McHugh, "Windows of vulnerability: A case study analysis," *IEEE Computer*, vol. 33, no. 12, pp. 52–59, 2000.
- [6] J. E. Y. Rossebø, M. S. Lund, K. E. Husa, and A. Refsdal, "A conceptual model for service availability," Research report 337, Department of Informatics, University of Oslo, 2006.
- [7] J. C. Laprie, Ed., *Dependability: Basic Concepts and Terminology*. Springer, 1992.
- [8] *ISO 7498-2, Information Processing Systems – Interconnection Reference Model – Part 2: Security Architecture*, International Standards Organization, 1989.
- [9] *ISO/IEC 17799, Information technology – Code of practice for information security management*, International Standards Organization, 2000.
- [10] *ISO/IEC 13335, Information technology – Security techniques – Guidelines for the management of IT security*, International Standards Organization, 2001.
- [11] A. Avižienis, J.-C. Laprie, and B. Randell, "Fundamental concepts of dependability," in *Third Information Survivability Workshop (ISW)*, 2000.
- [12] F. den Braber, M. S. Lund, K. Stølen, and F. Vraalsen, "Integrating security in the development process with UML," in *Encyclopedia of Information Science and Technology*. Idea Group, 2005, pp. 1560–1566.
- [13] M. S. Lund, F. den Braber, and K. Stølen, "Maintaining results from security assessments," in *Proc. of the 7th European Conference on Software Maintenance and Reengineering (CSMR)*. IEEE Computer Society, 2003, pp. 341–350.
- [14] *AS/NZS 4360:1999, Risk Management*, Standards Australia, 1999.
- [15] A. Avižienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, 2004.
- [16] J. F. Meyer, "Performability evaluations: Where it is and what lies ahead," in *Proc. of the International Computer Performance and Dependability Symposium*. IEEE Computer Society, 1995, pp. 334–343.
- [17] *UML 2.0 Superstructure Specification, formal/05-07-04*, Object Management Group, 2006.
- [18] J. Rosenberg, *A Presence Event Package for the Session Initiation Protocol SIP*, RFC 3856, 2004.
- [19] *Presence Service; Architecture and functional description, Stage 2*, Third Generation Partnership Project, Technical Specification Universal Mobile Telecommunications System (UMTS), 3GPP, TS 23.141 V 7.3.0 (2007-10), Release 7, 2007.
- [20] T. Ojala, J. Korhonen, M. Aittola, M. Ollila, T. Koivumäki, J. Tähtinen, and H. Karjaluoto, "SmartRotuuaari context-aware mobile multimedia services," in *Proc. 2nd International Conference on Mobile and Ubiquitous Multimedia*. Washington, DC, USA: IEEE Computer Society, 2003, pp. 9–18.
- [21] European Telecommunication Standardisation Institute, "Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN)," [http://portal.etsi.org/tispan/TISPAN\\_ToR.asp](http://portal.etsi.org/tispan/TISPAN_ToR.asp), 2008.
- [22] J. E. Y. Rossebø, S. Cadzow, and P. Sijben, "eTVRA, a threat, vulnerability and risk assessment method and tool for eEurope," in *ARES*. IEEE Computer Society, 2007, pp. 925–933.

## Appendix C – ICIN Paper

This appendix contains the reviewed academic paper *Combining IMS and Web 2.0 - Understanding the Availability* by Arlene Pearce, Terje Jensen, Judith Rossebø. This paper will be presented by myself at the International Conference on Intelligence in Networks (ICIN) 2008 in Bordeaux France, will be presented October 23, 2008.

# Combining IMS and Web 2.0 – Assessing the Service Availability

Arlene Pearce [pearce@pvv.ntnu.no](mailto:pearce@pvv.ntnu.no), Terje Jensen and Judith E. Y. Rossebø{[terje.jensen1](mailto:terje.jensen1@telenor.com), [judith.rossebo](mailto:judith.rossebo@telenor.com)}@telenor.com  
Telenor Research and Innovation, NO-1331 Fornebu, Norway

**Abstract**—Combining Web 2.0 and IMS has gained traction for several reasons. Drivers include users' interest in publishing content, innovation volume behind Web solutions and opening channels for commercial interests. In such an environment several components of a service should be orchestrated in order to provide a persistent user experience. Commonly a browser-like interface is then implemented and the interface could be divided into a set of regions. Then, certain parts of the service may work to the complete satisfaction of the user, while other parts are not quite up to the planned behaviour. It is usually also included as the planned behaviour that unauthorized users should not get access to the services or relevant data.

Availability requirements grow backed by increasing commercial interests and growing volume. Accessibility and exclusivity are the two essential characteristics of availability that need to be addressed. This paper presents an enhanced model for service availability intended to capture characteristics of orchestrated services. Application of this model is done by illustrating the combination of external Web 2.0 applications like Facebook and IMS Presence information in the IMS domain.

**Index Terms**—IMS 2.0, Web 2.0, Facebook, Flickr, del.icio.us, Availability concept, service orchestration.

## I. INTRODUCTION

WEB solutions are finding their growing place in the telecom environment. One example of this is the drive behind combining IMS (IP Multimedia Subsystem) and Web 2.0. Two appealing factors related to web solutions include the innovation volume and the modularity concept. This is also fuelled by the commercial and individual users publishing interests. Collating the user experience would imply that a number of service components should be orchestrated. In addition, individual user preferences should be respected. Those preferences would likely be distributed across several systems, within a provider's IMS servers, but also across servers maintained by partners.

Preparing for supporting agile services and overall cost savings, more and more telecom services migrate from dedicated networks to a common IP-based infrastructure. Keeping in mind key commercial and social importance of telecom, the IP-based infrastructure must support services complying with acceptable availability requirements.

Traditionally, the notion of availability has been defined as the probability that a system is working at time  $t$ , and the availability metric has been given by the uptime ratio,

This work has partially been funded by the Research council of Norway project SARDAS (152952/431) and partially within the Eureka project Mobile Fixed Convergence in Multiaccess Environments, Mobicome.

representing the percentage of time that a system is up during its lifetime [1]. Accompanying this interpretation, failure reporting procedures have also been described, e.g. [2] for Public Switched Telephony Network, PSTN. This understanding has served well for describing and analyzing availability of services delivered in dedicated networks such as for voice services in the PSTN/ISDN. However, for describing service availability characteristics and analyzing availability of services in the vastly distributed environment in which IP-based services are deployed, an enhanced notion of availability is required.

Considering the emerging range of IP-based services being delivered in public and private networks today, several challenges follow from the traditional understanding of availability. This paper addresses two challenges. The first challenge is that even with a high mean rate of availability, failure that occurs during peak service request periods will result in high operational loss. One such scenario is a web service with 99.999% average availability that loses connectivity for 5 minutes during peak sales of concert tickets. Such bursty behaviour patterns could be seen for several of the services [3]. The second challenge is that when presented with a set of service components, a user may have different expectations of quality for each component. One example is a buddy list with Presence information fed by various IMS Presence agents.

In the multi-application environment implied by IMS, several features may contribute to the overall user experience. For example, the user interface may collect parts of Presence information, location-dependent data, calendar tasks, streaming video and other service components. Different parts of the user interface may be updated by different servers. Hence, the user experience is collated from different sources. Moreover, the different parts of the user interface may have different weights in the experience depending on the user tasks.

We focus on the problem of considering the variability of the contributions of the different parts involved in the services to the overall availability of a service. Issues are the adequacy of the current availability concept, the definition of availability and the availability management.

The service availability concept model motivated and introduced in [4] is presented and exemplified by a case. This paper, building on [5], presents and elaborates on a conceptual model for service availability providing a case study to demonstrate the applicability of the model to service provisioning in a distributed IMS service environment.

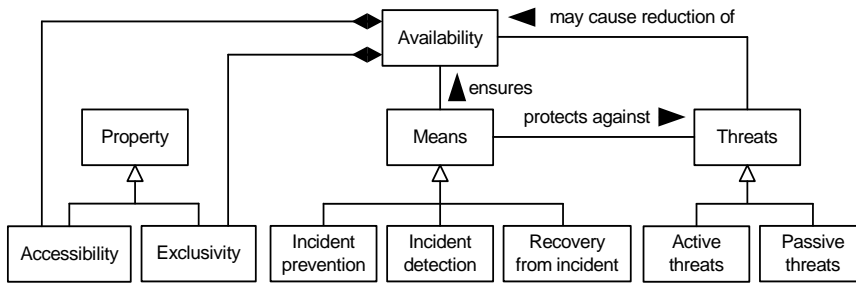


Fig. 1. Conceptual model for service availability

Sect. II provides a brief introduction to the enhanced service availability concept. Sect. III presents an overview of Presence service orchestration in IMS with external Web 2.0 Presence sources. Sect. IV exemplifies how the enhanced service availability concept can be applied for a federated Presence implemented across IMS and external servers.

## II. SERVICE AVAILABILITY

The setting for the enhanced service availability concept is derived from the fields of dependability and security. The definition of availability used as a basis for the enhanced service availability concept is: The property of being accessible and usable on demand by an authorized entity [6], [7]. This definition captures the integral part of securing availability by ensuring access to authorised users while also addressing the aspect of a service being usable in addition to the traditional aspect of readiness for correct service.

The notion of service availability has been further refined using this definition as a basis, to include addressing the *exclusivity* aspect of ensuring that a service is provided to the authorized users *only* [4]. This aspect is important because a system must know how many users are expected to access a service at a given time as well as how long the users are expected to access the service. If the means to ensure that authorized users *only* are accessing a service is too weak, and unauthorized users are able to access a service, the service availability for authorized users may be affected.

The conceptual model of dependability consists of three parts: the attributes of, the threats to and the means by which dependability is attained [8] and provides a basis for the service availability conceptual model as motivated in [9]. In order to classify threats to availability and means to achieve availability in a security setting, we are also motivated by the approach used in the security field of risk analysis as in [10].

This is because incidents resulting in loss of availability do not necessarily escalate into faults and therefore classification of means in terms of faults may become insufficient for availability analysis. An example is the hijacking of user sessions by an attacker or group of attackers, preventing the authorised user or group of users from accessing the service. This incident results in loss of service availability for a set of users, without incurring a fault in the system. An unwanted incident is defined in [11] as an incident such as

loss of confidentiality, integrity and/or availability. A fault is an example of an unwanted incident. The service availability conceptual model therefore classifies the means to achieve availability in terms of countering unwanted incidents.

Services can exist in numerous degraded but operational/usable/functional states between up and down or correct and incorrect. For example, an online newspaper may behave erratically with slow response times for displaying articles browsed without going down or becoming completely unavailable. This means that a more fine grained measure of availability is needed than pure up or down.

The enhanced notion of service availability encompasses both exclusivity, the property of being able to ensure access to authorised users only, and accessibility, the property of being at hand and useable when needed. Exclusivity involves ensuring that unauthorised users cannot interrupt, hijack, or prevent the authorised users from accessing a service. The focus is on preventing the denial of legitimate access to systems and services by prohibiting unauthorised users from interrupting, or preventing authorised users from accessing services. The aim is to ensure access to users while keeping unauthorised users out.

Accessibility is defined as the quality of being at hand and usable when needed. We divide accessibility properties into three major areas: timeliness, correctness and usability. Timeliness is the ability of a service to perform its required functions and provide its required responses within specified time limits. Usability is concerned with the user's perception of the service, and the ease of use of the service. The measure of correctness of a service may differ widely between different kinds of services. These considerations motivate a notion of service degradation [12]. Service degradation can be defined as reduction of service accessibility.

In summary, the overall conceptual model can be depicted as in Fig. 1 (illustrated in UML 2.x format [13]). Availability is affected by means and threats. Means can ensure availability by protecting against threats. Threats may lead to unwanted incidents which may cause reduction of availability.

By means to ensure availability we address protection of the service from incidents leading to a loss of availability. We have categorized the means into i) incident prevention: how to prevent incidents causing loss of availability (e.g. access

control, integrity protection ensuring graceful degradation); ii) incident detection: how to detect incidents leading to loss of availability (e.g. traffic inspection, audit logs); and, iii) recovery from incident: the means to recover after an incident has lead to a loss of availability (e.g. system adaptability, robustness, maintainability, redundancy).

Threats may originate on the inside (inside attackers) or the outside (outside attackers) of the system. The impact of threats varies with the nature of the threats; some threats may result in degradation of the service, others in complete loss of service. For the full motivation and explanation of the model, see [9].

Based on the conceptual model, the availability of a service can be analyzed with respect to exclusivity and accessibility aspects. On an abstract level, a mathematical representation can be given as follows; Let  $A$  denote a service with an availability property for a user group  $U$ , and let  $X$  denote the availability metric for service  $A$ . We represent  $X = (x_1, \dots, x_n)$  as an  $n$ -tuple where  $x_i$  is a measure of an aspect of availability. These include behavioural, preventive and correctness aspects. By this we mean that  $x_i$  describes requirements for a particular availability aspect. The minimum requirement for each  $x_i$  must be satisfied in order to fulfil the total availability requirement  $X$ . Using the conceptual model this idea can be refined as follows: We represent  $X$  as a tuple  $X = (X_1, X_2)$  where  $X_1$  measures the exclusivity properties, and  $X_2$  measures the accessibility properties. Essentially, the aim is to describe the degree of accessibility and exclusivity that is sufficient for the user to be able to activate and use the service. The purpose of service availability metrics is to measure how well service availability requirements have been met.

### III. PRESENCE ORCHESTRATION: IMS AND WEB 2.0

IMS has been promoted by several international bodies as a future platform for providing rich multimedia services. It is access agnostic in the sense that services could be provided over any access type and to any device, as long as the device is capable of supporting the proper client behaviour.

#### A. IMS principles

A layered architecture has been applied for defining IMS, as depicted in Fig. 2. In the core part, we find common session control and common user data. In IMS terms these are referred to as Call Session Control Functions (CSCF) and Home Subscriber Server (HSS), respectively. There are three types of CSCF supporting roaming users, interconnecting between domains and emergency sessions, but these are not depicted individually in Fig. 2.

The HSS stores user identities and user profiles. Each user profile contains at least one service profile with customised information pertaining to which applications are to be invoked, in which order and how services are to be executed. This capability for user-customised service mashups implies that IMS can be considered to be a user-oriented architecture. This is an important factor for combining IMS services with existing 3rd party Web 2.0 services.

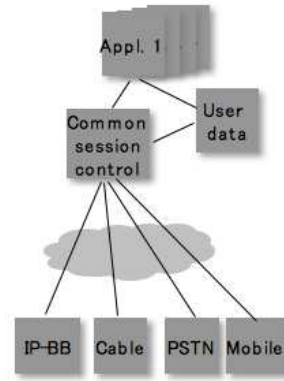


Fig. 2. Layered architecture of IMS (Note that clients are not illustrated)

A range of applications can reside in the common IMS core. Examples of application types are group list managers, Centrex, location, handover support.

#### B. Presence Service

One application that we analyse with respect to the service availability model presented above, is the Presence service. A Presence service is a system that accepts and stores Presence information from information providers, called Presentities and distributes this information to interested parties, called Watchers as illustrated in Fig. 3. Presence sources are nodes reporting Presence information. Examples of Presence sources are clients in handsets, mail/calendar servers, network elements and indications given through web portals. Presence information, as defined by [14], conveys the ability and willingness of a user to communicate across a set of devices. IMS uses SIP-based specifications to provide Presence functionality.

The Presence Server (PS) is located in the presentity's home network. It commonly includes both logic and data storage. Key capabilities include collecting, composing and filtering Presence information.

As depicted in Fig. 3, it is not only user status that can be reported through this mechanism. Other examples include changes in stock prices and programme information from radio stations. The different Presence items may be updated from different sources, involving different service providers (companies and individuals). This implies that the different providers must interact, and cooperate. An immediate case is the incorporation of Facebook news feeds ([www.facebook.com](http://www.facebook.com)), Flickr ([www.flickr.com](http://www.flickr.com)) photo album updates and del.icio.us ([del.icio.us/about/](http://del.icio.us/about/)) bookmark updates into an advanced IMS-based buddy list handler. This scenario would also include Presence items internal to IMS such as buddies with IMS subscriptions. Fig. 5 gives a logical view of the Presence entities in such a system from an IMS perspective. These particular third party services were chosen because they are popular Web 2.0

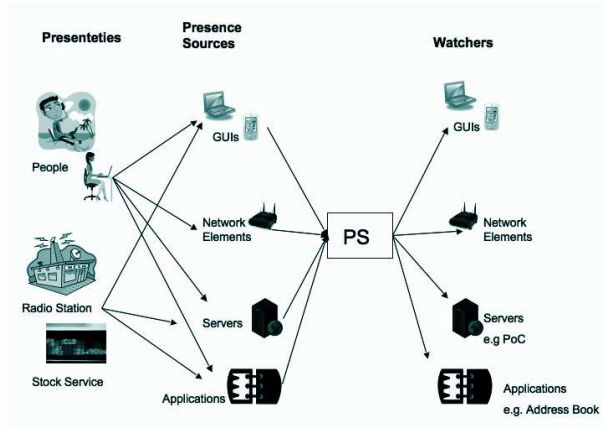


Fig. 3. Presence Server conveying information about users to defined Watchers

services. In addition these services expose HTTP interfaces, making it possible for entities in the IMS service domain to consume them. The various Presence agents presented here are described in 3GPP's Presence specification [15].

Subscription authorization policy is also provided by the PS. This controls which Watchers are authorised to view Presence information. Level of detail may differ for the different Presence items. For example, for certain buddies (for example the user's minor children), the user is allowed to see their locations, while for others (for example business contacts) location may be kept private. Again, these rights may vary over time (day, night, work, vacation etc.)

Both the accessibility and the exclusivity aspects of service availability discussed in Sect. II are related to the Presence item list. For each of the different groups in the item list, the degree of exclusivity and accessibility achieved may differ. For example, during leisure time, colleagues may not want updates on a user's whereabouts. On the other hand, this information becomes relevant during office hours.

A time stamp field in the message format may be used as a rudimentary security mechanism to prevent replay attacks (launched for example to capture user credentials for unauthorized access to a service). For example, tuples whose time stamp are older than the time stamp of the most recently received Presence document should be discarded.

#### IV. CASE – ENHANCED PRESENCE SERVICE AVAILABILITY

The enhanced Presence service based on IMS is assumed to combine a set of different features, such as location information as well as advertisements. Multiple levels of availability are particularly relevant when buddies are subscribers of other domains.

For such a multi-provider case, a block diagram could be constructed for each of the items in the list. For this configuration, certain details regarding implementation of Presence services in other domains would likely not be

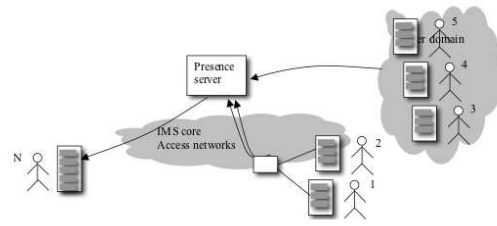


Fig. 4. User with buddies in different provider domains

available. This implies that accessibility levels would then be part of the Service Level Agreements (SLAs) established between providers. It might also be more suitable to state different levels towards user  $N$  for the different user groups in this configuration. However, this will probably depend on the SLA terms. An index  $j$  indicates the group of buddies:

$$A_j = \sum_i (A_{ij} \cdot \alpha_{ij}) / \sum_i \alpha_{ij}, \text{ where } \alpha_{ij} \text{ is the level of importance associated with user } i, \text{ and } A_{ij} \text{ is accessibility of Presence information for each user } i \text{ in group } j.$$

Fig 6 illustrates how availability of external Presence services can be modelled from the system depicted in Fig 5 with such a block diagram. Note that only the buddy list service from each provider is modelled Facebook (fb) friends, Flickr (fl) contacts and del.icio.us (dl) networks. Fig 5 indicates that other service modules are offered from these external providers but as they are loosely coupled it is possible to choose only a subset of services at any given time. It is also possible to model the availability of all the services.

In some cases, there may not be any pre-established SLA between the providers. It then becomes a business risk evaluation whether a service provider wants to state any performance levels in the service description to user  $N$ . Potentially, there may be differentiated levels for the different groups,  $j$ .

Level of importance related to a user,  $\alpha_{ij}$ , may vary depending on the role of the user,  $N$ . For example, during working hours, it is more important to follow work colleagues, for example, involved in the same project, than outside working hours. As projects come and go, the colleagues involved will also vary, requiring that this information is easily updated frequently.

##### A. Parameters Included in Enhanced Presence

The parameters given for a Presence item may include nickname, mode, location, as well as others. Typically, location could be given with different levels of granularity. In some cases, e.g. during roaming, the location may also be unknown. On the other hand, there are certain usages where location is considered prioritised.

For user  $N$ , different levels of importance could then be

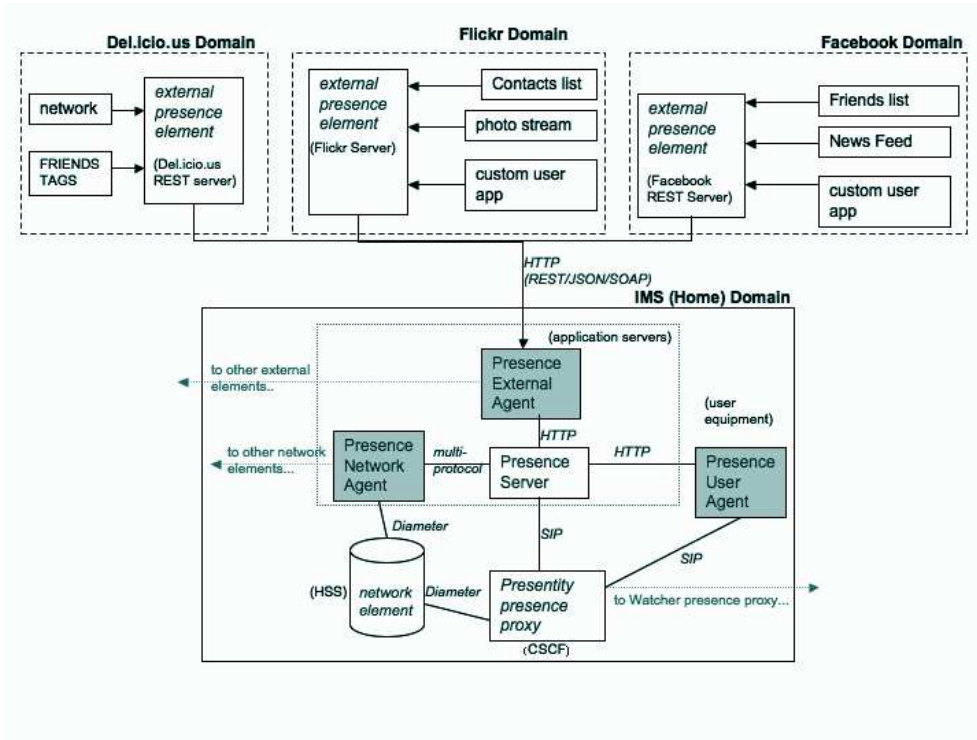


Fig. 5. External Web 2.0 Presence Items

attached to the different parameters. These importance weights may also differ for the different Presence items.

As these parameters may be pushed or pulled from different sources, different response or delivery times would result. In some respects, this is similar to the design of a web page consisting of a set of objects. In order to improve QoS and network performance, Presence parameters should be delivered in appropriate sub-groups. That is, waiting for the last Presence parameters before sending an update to the user's buddy list would likely result in delayed response times and degraded Quality of Experience.

This sub-grouping is particularly a critical aspect when information is collected from vastly different sources, some residing in semi-real-time environments while others within best-effort environment. In effect, this balances the different aspects of availability (correctness, timeliness and usability) as described in Sect. II.

### B. Use of Presence Items for Other Purposes

Having configured means for controlling Presence items on a terminal, one could utilise this for other purposes as well, such as advertising and time-related special offers. One such implementation has been tested in a real user environment in Finland, in the SmartRotuuaari project [16]. The service implemented included highly personalized direct marketing to customer's mobile phones.

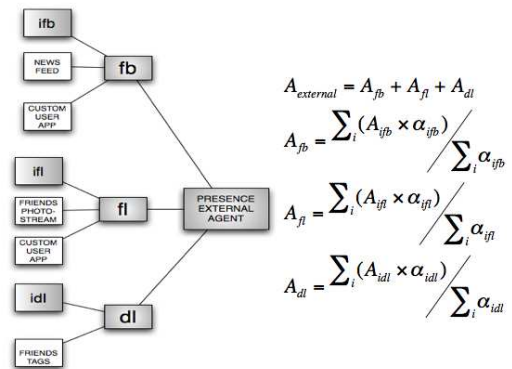


Fig. 6. Block Diagram for External Presence

When users accept such commercial activities, a service provider would then likely have an agreement with other companies for delivering the advertisements/offers. Then, there will also be requirements on accessibility for providing this feature. In the spirit of web 2.0, users could be given the ability to give reviews on advertised products they have tried. Watchers could then choose only to filter which advertisements they accept (for example: "only with a rating above 610" or "only products recommended by my buddies" or "only

products reviewed by more than 500 users”). Again, block diagrams could be made assisting the accessibility evaluation.

### C. Exclusivity

Overall exclusivity could be analysed in a similar manner as for accessibility. That is, either looking at the overall average or looking at each Presence item individually. However, the analysis approaches would likely differ as there are often other types of threats affecting the exclusivity aspect.

The aim with respect to the exclusivity aspect is to ensure that authorised users only should have access to Presence items that they subscribe to. Allowing unauthorised users to access Presence items may have a negative effect on accessibility, but also, may be in violation of privacy directives in the jurisdiction.

Commonly, for the Presence service there will be stricter requirements regarding exclusivity than for accessibility. This is mainly due to privacy aspects, avoiding any third party being able to follow a user’s actions. However, it is also important to ensure that the authorised users are not prevented or interrupted from accessing the Presence items. Activity initiated by unauthorised users can adversely affect the accessibility aspects.

### D. Threats and Corresponding Means

Considering the different network elements, protocols, data, etc. that are involved, various threats would be relevant. Examples of threat agents are malicious users, rogue service providers and unauthorised users masquerading as authorised users. It is important to note here that in the Web 2.0 context, service providers may also be individual users of the system. The vulnerability of the Presence service to distributed denial of service attacks should be evaluated.

Hence, combinations of means to address the threats, covering protection, detection and recovery will be required. Essentially, preventative mechanisms will try to eliminate the possibility of attacks by threat agents or to enable the Presence service nodes to be able to endure attacks without denying service access to authorised users. Detection and recovery will involve detecting attacks on nodes of the service, or against specific users, and responding immediately to restrict impact.

ETSI TISPAN has developed a threat vulnerability and risk assessment (*eTVRA*) method and tool that may be used to analyse the Presence service [17]. Using the *eTVRA* method and tool, the threats to availability of the Presence service can be analysed and a set of recommended countermeasures can be identified that when implemented will reduce the overall risk.

## V. CONCLUSION

Recognising the business of delivering dynamic information from widely different sources to individual users, orchestrating services become a key aspect. One example is given in this paper showing how Presence information can be collated from various internet sources and blended in IMS. Considering the federated nature of the Presence service, a range of challenging

aspects need to be addressed, including differentiation of Presence items and parameters for an item while also handling multiple sources of Presence data.

This paper outlines a service availability model, applied for the Presence service in an IMS context. A key point is to include both the accessibility and the exclusivity aspects of the service availability measure. Hence, only the authorized users should be ensured access to the service, and with the proper service levels. So far, it seems that exclusivity is an aspect of availability that has rarely been included in the literature. However, the concept presented here shows where exclusivity fits in with an enhanced notion of service availability.

The enhanced notion seems even more important when considering collaboration between providers, and also between different roles of the same user. Proper service orchestration, then becomes even more important and more challenging.

## REFERENCES

- [1] S. M. Ross, *Introduction to probability models*, 6th ed. Academic Press, 1997.
- [2] P. Enriquez, A. B. Brown, and D. A. Patterson, “Lessons from the PSTN for dependable computing,” Workshop on Self-Healing, Adaptive and self-MANaged Systems (SHAMAN), 2002.
- [3] D. Clark, W. Lehr, and I. Liu, “Provisioning for bursty Internet traffic: Implications for industry and Internet structure,” MIT ITC Workshop on Internet Quality of Service, 1999.
- [4] J. E. Y. Rossebø, M. S. Lund, K. E. Husa, and A. Refsdal, “A conceptual model for service availability,” *Quality of Protection: Security Measurements and Metrics*, vol. 23, 2006.
- [5] J. E. Y. Rossebø, A. Pearce, and T. Jensen, “On understanding availability of services based on ip multimedia subsystem,” in *18th ITC Specialist Seminar – Quality of Experience*, to appear.
- [6] *ISO 7498-2, Information Processing Systems – Interconnection Reference Model – Part 2: Security Architecture*, International Standards Organization, 1989.
- [7] *ISO/IEC 13335, Information technology – Security techniques – Guidelines for the management of IT security*, International Standards Organization, 2001.
- [8] A. Avižienis, J.-C. Laprie, and B. Randell, “Fundamental concepts of dependability,” in *Third Information Survivability Workshop (ISW)*, 2000.
- [9] J. E. Y. Rossebø, M. S. Lund, K. E. Husa, and A. Refsdal, “A conceptual model for service availability,” Research report 337, Department of Informatics, University of Oslo, 2006.
- [10] F. den Braber, M. S. Lund, K. Stølen, and F. Vraalsen, “Integrating security in the development process with UML,” in *Encyclopedia of Information Science and Technology*. Idea Group, 2005, pp. 1560–1566.
- [11] *AS/NZS 4360:1999, Risk Management*, Standards Australia, 1999.
- [12] J. F. Meyer, “Performability evaluations: Where it is and what lies ahead,” in *Proc. of the International Computer Performance and Dependability Symposium*. IEEE Computer Society, 1995, pp. 334–343.
- [13] *UML 2.0 Superstructure Specification, formal/05-07-04*, Object Management Group, 2006.
- [14] J. Rosenberg, *A Presence Event Package for the Session Initiation Protocol SIP*, RFC 3856, 2004.
- [15] *Presence Service; Architecture and functional description, Stage 2*, Third Generation Partnership Project, Technical Specification Universal Mobile Telecommunications System (UMTS), 3GPP, TS 23.141 V 7.3.0 (2007-10), Release 7, 2007.
- [16] T. Ojala, J. Korhonen, M. Aittola, M. Ollila, T. Koivumäki, J. Tähtinen, and H. Karjaluoto, “SmartRotuaari context-aware mobile multimedia services,” in *Proc. 2nd International Conference on Mobile and Ubiquitous Multimedia*. Washington, DC, USA: IEEE Computer Society, 2003, pp. 9–18.
- [17] J. E. Y. Rossebø, S. Cadzow, and P. Sijben, “eTVRA, a threat, vulnerability and risk assessment method and tool for eEurope,” in *ARES*. IEEE Computer Society, 2007, pp. 925–933.