

Dynamic Service Orchestration in the IP Multimedia Subsystem

Richard Spiers

Supervised By:
Neco Ventura



Thesis Presented for the Degree of
DOCTOR OF PHILOSOPHY
in the Department of Electrical Engineering
UNIVERSITY OF CAPE TOWN
March 2017

The copyright of this thesis vests in the author. No quotation from it or information derived from it is to be published without full acknowledgement of the source. The thesis is to be used for private study or non-commercial research purposes only.

Published by the University of Cape Town (UCT) in terms of the non-exclusive license granted to UCT by the author.

Declaration

I declare that the above thesis is my own unaided work, both in concept and execution and that apart from the normal guidance from my supervisor, I have received no assistance except as stated in the text of this document. This work is being submitted for the Doctor of Philosophy Degree in Electrical Engineering at the University of Cape Town. Neither the substance nor any part of the thesis has been submitted in the past, or is being, or is to be submitted for a degree at this university or at any other university.

Acknowledgements

The completion of this dissertation has been a long journey. One does not know the road that one's life will take, and there have been several more turns than expected since I began my post graduate studies. Fortunately, I have been blessed with support and encouragement from many different sides throughout the process.

To Neco Ventura - Thank you for your supervision, both during my undergraduate degree as well as my post graduate studies. I would not be writing this page without your guidance in key points through out the last decade. Thank you and the centre for the assistance in presenting my work overseas, and for granting me opportunities few people receive.

To the members of the UCT Communications Research Group, especially David Waiting, Richard Good and Vitalis Oziyani - Thank you for your continual assistance, right from my under graduate thesis project all the way through to my PhD. Without your inspiration it is doubtful whether I would have continued my studies. You welcomed me into the lab and taught me the ropes. My time as a post graduate student would have been much less enjoyable without your contributions.

To my parents-in-law - Thank you for all the support, especially my late father-in-law Brian McCarthy. Thank you for never losing faith in my ability to complete this work. For your quiet enquiries about how my project was progressing, kind encouragements and unwavering faith. I know how happy this day would have made you.

To my parents, John and Sandra Spiers - Your love and support throughout my life has always pushed me to new heights. Thank you for the sacrifices you made that have made me the man I am today. I count myself lucky to have been blessed with you as my parents.

Finally, to my wife, Belinda Spiers - I owe you a huge debt of gratitude. For missing out on countless events and holidays in order to support my work, for putting up with the many hours spent in my study far from friends and family, and for covering for me when I was missing in action at family gatherings. Thank you for always providing me with the support I needed to complete this work. A new chapter in our life begins with the completion of this dissertation, and I can't wait to share it with you.

Abstract

The continued growth of the Internet has resulted in a massive uptake of different communication services. An ever increasing number of users are utilising rich multimedia services, and are both consuming and producing user generated content. This rapid growth in bandwidth and complexity has resulted in challenges for the previous architectures used in telecommunication networks. As a result, two of the major challenges faced by telecommunication network operators are an increase in the number of services that an end user expects to be able to use, as well as the desire to have personalised, custom use of these services. Over-the-top providers have begun to offer telecommunication services over the physical infrastructure of the traditional network operator, bypassing the lengthy and costly development cycle involved in offering carrier grade communication services.

To adapt to this new landscape, the IP Multimedia Subsystem (IMS) was proposed. Its ongoing standardisation is being led by the 3rd Generation Partnership Project (3GPP) and it is a Next Generation Network (NGN) that is heading towards converged multimedia services. This NGN contains IP based network services that allow telecommunication operators the reuse of common building blocks for new services, reducing the time and cost to provision a new service.

The thesis begins with an overview of the current landscape that plays a critical role in the formation and deployment of the IMS. This is followed by a literature review of service provisioning in the IMS. It looks at both the current, standardised architecture as well as several new approaches currently being researched. It determines that there is a need for further work in this area, and this concurs with the view of the standards body responsible for the IMS. However, there are limits on what can be proposed moving forward as deployment of the IMS architecture has already begun.

The work continues with the creation of a novel architecture that would allow a network operator greater flexibility in routing service requests, as well as the ability to expose this functionality to the end user. This allows the end user to customise their service usage, something which is in great demand with the rise of user generated content and the new services being used over the internet.

A prototype of the architecture is evaluated to determine the feasibility of the proposed architecture. Emphasis is placed on ensuring correct operation and increasing the range of functionality available.

The prototype is evaluated in a test-bed provisioned through the use of Amazon's Elastic Compute Cloud (EC2). This is an Infrastructure as a Service (IaaS) offering that provides resizeable or elastic computing resources to researchers or developers, allowing the rapid deployment and reconfiguration of many different virtual servers.

Different combinations of services that were identified as not being possible under the previous standardised architecture are evaluated in this test-bed. The proof of concept implementation provides the necessary functionality to execute these service combinations successfully without adding custom combined services. The evaluation is comprised of both functional and performance tests although the emphasis is on the former. As all components have been made available under open source licenses, it is possible for other researchers to reproduce this work and continue investigating this area. The work successfully demonstrates improved service triggering in the IMS whilst providing end users a much higher degree of control over the use of their telecommunication services.

Contents

1	Introduction	1
1.1	Non traditional communication services	2
1.1.1	App Stores	2
1.1.2	Increase in smart devices	3
1.1.3	VoIP	4
1.1.4	Messaging	5
1.2	User Centric Computing	6
1.2.1	Social networking	6
1.2.2	Content creation	7
1.2.3	Google Voice	7
1.3	Next Generation Networks	8
1.3.1	Principles	9
1.4	The IP Multimedia Subsystem	11
1.4.1	Services	12

1.4.2	Signalling	12
1.5	Service Orchestration	13
1.5.1	Feature Interaction	13
1.5.2	Contextual Service Triggering	15
1.5.3	Service registration	16
1.5.4	User centric service control	16
1.5.5	Service Triggering in the IMS	16
1.6	Thesis Objectives	17
1.6.1	Dynamic Service Orchestration	19
1.6.2	Service Registration and Monitoring	19
1.6.3	User Policy Creation and Modification	20
1.7	Thesis Scope and Limitations	20
1.8	Contributions	21
1.9	Thesis Outline	24
2	Application Triggering Architecture in the IMS	25
2.1	Current Functionality	25
2.1.1	Architecture	26
2.1.2	Service Triggering	29
2.1.3	Discussion	34
2.2	Impacts of Service Brokering - 3GPP Study	35

2.2.1	Motivation for improvements	35
2.2.2	Definitions	37
2.2.3	Requirements	39
2.2.4	SCIM Architecture	40
2.2.5	Reference architecture	42
2.2.6	Functionality	43
2.2.7	3GPP Conclusions	44
2.2.8	Discussion	45
3	Related Work	48
3.1	Service Interaction Management	49
3.1.1	Information required	50
3.1.2	Service conflict detection	51
3.1.3	Specific service combinations	54
3.1.4	State Tracking mechanisms	56
3.2	Reducing load and session setup delay	57
3.3	SCIM Architecture	61
3.3.1	Internal as part of AS	61
3.3.2	Internal as part of S-CSCF	62
3.3.3	External as stand alone component	62
3.4	Context and context aware applications	63

3.4.1	SIMPLE Presence	63
3.4.2	Presence Architecture in the IMS	66
3.4.3	Example context aware services	68
3.5	Discussion	69
4	Proposed Design	71
4.1	Design Considerations	71
4.1.1	Design requirements	72
4.1.2	Core component integration	72
4.1.3	Service components	73
4.1.4	Client components	73
4.2	New functionality	74
4.2.1	Information Gathering	74
4.2.2	Dynamic Service Routing	75
4.2.3	User customisation	77
4.2.4	Efficient control signalling	77
4.2.5	Service description language and service control	78
4.3	New Architecture	81
4.3.1	Service Capability Interaction Manager	81
4.3.2	Context Server (CS)	84
4.3.3	Service Repository Server (SRS)	87

4.3.4	User Service Policy Server (USPS)	88
4.4	Discussion	90
5	Implementation	92
5.1	Implementation considerations	92
5.1.1	Platform and Language Choice	93
5.2	SIP Library	94
5.3	Client	95
5.3.1	Signalling	96
5.3.2	Registration	97
5.3.3	User context	98
5.3.4	Multimedia	98
5.3.5	Services	98
5.4	Framework Components	99
5.4.1	Context Server (CS)	99
5.4.2	Service Repository Server (SRS)	99
5.4.3	User Service Policy Server (USPS)	100
5.4.4	Service Capability Interaction Manager (SCIM)	105
5.5	Discussion	107
6	Evaluation Platform	108
6.1	Test-bed environment	108

6.1.1	Supporting infrastructure	110
6.1.2	A note on performance	110
6.2	Test-bed components	112
6.2.1	Call Session Control Functions	112
6.2.2	Home Subscriber Server	113
6.2.3	Presence Subsystem	114
6.3	Test-bed functionality	115
6.3.1	Registration	115
6.3.2	Standardised Application Triggering Architecture . . .	115
6.4	User equipment	119
6.4.1	IMS client	119
6.4.2	Web browser	121
6.5	Measuring and utility software	122
6.5.1	Smokeping	122
6.5.2	Wireshark	123
6.5.3	Call-flow Sequence Diagram Generator	123
6.5.4	IMS Bench / SIPp	124
6.6	Discussion	126
7	Test Results	127
7.1	Functional Testing	127

7.1.1	Original Functionality	128
7.1.2	Enhanced Functionality	131
7.1.3	Discussion	141
7.2	Performance Testing	143
7.2.1	Registration	144
7.2.2	Standard call setup delay	151
7.2.3	Discussion	157
8	Conclusions and Recommendations	158
8.1	Conclusions	159
8.1.1	IP based service competition	159
8.1.2	User Centric Computing	160
8.1.3	Standardised architecture	160
8.1.4	State of art	161
8.1.5	Requirements for SCIM architecture	162
8.1.6	New functionality developed	163
8.1.7	Conflict Detection	164
8.1.8	Service Registration	164
8.1.9	User Controlled Customisation	165
8.1.10	Contextual Routing	166
8.1.11	Cloud based test-bed	166

8.1.12	Session Setup Delay	167
8.2	Future work	168
8.2.1	Optimised implementation	168
8.2.2	Service conflict resolution	169
8.2.3	Machine learning	169
References		171
Appendix A: Example Test Report		190

List of Tables

3.1	Main areas of related work.	49
3.2	Comparison of related work.	70
4.1	Requirements versus added functionality.	91
7.1	CPU usage - Registration over standard architecture.	146
7.2	CPU usage - Registration over proposed architecture.	146
7.3	Free Memory - Registration over standard architecture.	148
7.4	Free Memory - Registration over proposed architecture.	148
7.5	Delay for REGISTER over standard architecture.	150
7.6	Delay for REGISTER over proposed architecture.	150
7.7	Delay for INVITE over standard architecture.	153
7.8	Delay for INVITE over proposed architecture.	153
7.9	Total session delay over standard architecture.	154
7.10	Total session delay over proposed architecture.	154

List of Figures

2.1	Standardised IMS architecture for service triggering.	27
2.2	Evaluation of iFC in the IMS.	34
2.3	Service Chaining in the IMS.	34
2.4	Centralised SCIM architecture.	41
2.5	Distributed SCIM architecture.	41
2.6	Hybrid SCIM architecture.	42
2.7	3GPP proposed reference architecture for SCIM.	43
3.1	IMS Presence Signal Flow.	68
4.1	Service Configuration XML.	80
4.2	Architecture for proposed framework.	82
4.3	Context Server Subscribing to End User Context.	86
4.4	Example signalling flow for service monitoring.	88
5.1	Abstract methods and events of the SIPApp Class.	95
5.2	Debug Log showing SIP signalling.	96

5.3	Client modules.	97
5.4	Screen shot showing options presented to end user.	102
5.5	Service flow scenario customised by end user.	103
6.1	Screenshot showing web interface to FHoSS.	114
6.2	OpenXCAP and OpenSIPS based presence system.	116
6.3	Client Registering with Fokus OSIMS.	117
6.4	Presence AS configured in FHoSS.	118
6.5	TP redirecting requests to SCIM.	118
6.6	IMS Client messaging functionality.	120
6.7	Wireshark GUI with a SIP packet capture.	123
6.8	Extract from IMS Bench XML configuration file.	125
7.1	Signal diagram showing client IMS registration.	129
7.2	Signal diagram showing client IMS PUBLISH request.	129
7.3	Signal diagram showing client IMS SUBSCRIBE request.	130
7.4	Signal diagram showing voice call.	132
7.5	SIP INVITE request with SDP for video call.	133
7.6	SIP INVITE redirected to call metadata log service.	134
7.7	Trigger Point redirecting certain requests to SCIM.	135
7.8	Static redirect through contact differentiation iFC.	136
7.9	Creating a contact specific redirection through the USPS.	137

7.10	Contact differentiation redirection via SCIM.	138
7.11	Time based service routing.	140
7.12	SIP Signal Flow for 3GPP scenario.	142
7.13	Average CPU usage during registration.	147
7.14	Average Free Mem usage during registration.	149
7.15	Mean delay during first registration exchange.	151
7.16	Comparison of session establishment delay.	155
7.17	Increase in session setup delay with increased load.	157

Chapter 1

Introduction

The ability for end users to communicate with each other has increased tremendously since the introduction of the Internet. The past decade has seen a large range of services introduced over the Internet which allow end users to communicate with others through a variety of means. These services are generally known as “Over The Top” (OTT) services as they are offered on and above the carriers network. While very positive for the end user, these changes have introduced difficult challenges to the existing telecommunication operators. Traditional telecommunication operators have not been able to innovate at the same rate as internet based start-ups for various reasons, including the carrier-grade nature of their products, the length of various standardisation procedures, the capital needed for infrastructure etc.

This has resulted in a markedly changed telecommunication landscape which is very different to the scenario for which previous generations of networks were designed. Some of the notable improvements and challenges here, as well as the necessary background for the work contained in this dissertation, are laid out below. This is followed by a discussion detailing the objectives, scope and limitations of the thesis. The chapter is concluded with the major contributions of this work together with an outline for the remainder of the dissertation.

1.1 Non traditional communication services

The introduction of the Apple iPhone marked a specific turning point in the Internet revolution. Carriers previously played a much larger role in determining what services or feature would be available on their networks. The introduction of the iPhone by Apple, a computer software and hardware company, was a very disruptive action to the traditional telecommunications ecosystem. This, together with the other developments described below, revolutionised the telecommunication landscape and vastly increased the number of services available on an end user's smart phone.

1.1.1 App Stores

The release of the Apple iPhone was followed shortly by the Apple App Store, which resulted in the globalisation of the concept of an “*App Store*”. App Stores allow developers across the world access to the end user via their telecommunication device, breaking the monopoly that carriers used to enjoy. The Apple App Store has now seen over 50 billion downloads as of May 2013, with more than 800 applications being downloaded every second on average [1].

The Android equivalent, Google Play, has seen over 48 billion application installations, with 900 million device activations as announced at the Google I/O conference in May 2013. These are examples of the most famous App Stores, but there are a large range of them from many different companies. No single company has the resources to compete against this number of developers and applications. The network operator can no longer compete as a sole provider of telecommunication services on mobile devices.

1.1.2 Increase in smart devices

In terms of the number of actual devices in use, we have more than 1.5 million Android “activations” per day with one billion android phones expected to be sold by the end of 2013 as announced by Eric Schmidt (Executive Chairman of Google) at the Dive Into Mobile conference in April 2013. The world wide percentage of smart phones was sitting at around 30 % of the global mobile market (1.5 billion out of 5 billion plus) in 2013 [2]. A survey of 40 countries conducted in 2015 revealed that a global median of 43 % say that they own a smartphone, climbing to as high as 88 % in advanced economies such as South Korea. Additionally, emerging countries are seeing double digit growth year on year for smartphone ownership, and several countries such as Turkey, Malaysia, Chile and Brazil have seen increases of over 25 percentage points since 2013 [3].

These new devices allow third party developers to offer new services (including communication services) directly to the end user, bypassing the traditional telecommunication operator. It allows them to innovate rapidly as there is very low cost to establish a start-up with minimal business capital required. The telecommunication operators used to hold dominion over both messaging and voice communication, but both of these functions are now exposed through a myriad of providers (e.g. Skype, Viber, WhatsApp, WeChat etc.). The network operator is excluded from making profit from these services other than via data or breakout fees. Revenue from the telecommunication operator’s traditional services are mainly flat or in some cases falling [4, 5]. The prices for individual services are decreasing, and non traditional telecommunication services offered by third party providers are experiencing rapid growth [6].

1.1.3 VoIP

Voice over IP (VoIP) has now been firmly established in both the minds and habits of end users, reducing the value of pure voice services from telecommunication operators. Skype now has in excess of 280 million users [7], and as of April 2013, they were on average using 2 billion minutes a day [8]. This rose to 3 billion minutes a day by 2015 [9]. TeleGeography estimates that Skype's international on-net traffic grew by 51 billion minutes in 2012 which is "more than twice [the volume growth experienced] by all the telecommunications operators in the world combined" [5].

Another popular voice client is Viber, which has seen growth from 200 million users in 2013 to over 608 million users in 2014 [10, 11]. Skype and Viber are only two examples of many more international VoIP providers that provide voice communication applications for mobile phones.

Tackling this area from a different angle is Twilio, which is a cloud computing company that develops a programmatically controllable API to expose VoIP functionality. Twilio does not act as a VoIP service provider to the end user, but rather allows other companies to build VoIP into their products. Twilio has more than 150 000 developers using this service and has made more than 500 million phone calls as announced during their developers conference in October 2012 [12]. While Twilio still uses traditional infrastructure as the last mile to reach the target user's device (e.g. their existing fixed or mobile number), the end user placing the call is doing so through another application or service which is not provided by the network operator.

This leads to a loss of revenue as well as reduction in brand value and brand awareness for the network operator who no longer has sole control of the end user's telecommunication services. Twilio has close to 30 000 customer accounts who are utilising their platform, bringing in over 130 million dollars (USD) in revenue for Twilio [13].

1.1.4 Messaging

The Short Message Service (SMS) has been a very profitable revenue stream for the network operators. However, many users are now opting to utilise data based message services such as WeChat, WhatsApp Messenger or iMessage instead of SMS. This is because it is often cheaper as well as being more convenient. iMessage will send a message across IP connectivity without input from the end user if it is possible, falling back to SMS should data connectivity not be available. The service is also enhanced with enriched functionality, such as detailed delivery notifications.

A popular Android based messaging application, WhatsApp Messenger, was handling more than 20 billion messages a day [6]. According to Informa, by the end of 2013, OTT messaging traffic will total 41 billion messages per day compared with 19.5 billion SMS messages per day. As these new applications and services become more popular the network operator's services are used less and less. For example there was a 22% decrease in the number of SMS messages sent on Christmas Eve for 2011 on one operators network in Finland [4]. At Facebook's F8 developer conference in 2016, Mark Zuckerberg announced that Facebook messenger and WhatsApp see 60 billion messages sent daily versus 20 billion for SMS. This was later reported in their first quarter earnings call in the same year [14].

Network operators are beginning to need new business models to profit off these third party services as they slowly replace their own. These alternative messaging services or applications are not limited only to smart phone devices. 2Go Messaging is a South African developed service which heavily targets feature phones (i.e. non smartphones). They have over 50 million registered users, mainly in Nigeria, Kenya and South Africa, supporting over 1500 different types of feature phones in addition to their recently launched smart phone versions [15].

1.2 User Centric Computing

Another large change that the Internet has been responsible for is the concept of user centric computing. End users are now accustomed to sharing large amounts of information on the web. They use services that allow them to express themselves to a large audience, whether it is friends and family or a global audience. Many web services now act as communication facilitators rather than providing value by themselves.

1.2.1 Social networking

A prime example of these user centric services are the social networking services. Various different social networking services have become standard services for the average end user. As of March 2016, Facebook has over 1.65 billion monthly active users, which increased at a rate of 15% year-over-year. Of these users, at least 1.51 billion use mobile devices to access Facebook services. More than half of these users login every day [16]. Of particular interest is the fact that they have more than 1 billion users for their messaging service, Facebook Messenger [17].

Another challenging aspect being created by social networking services is one of authentication. One of the key values for a network operator is the fact that they have strict authentication measures that can track service usage to a particular mobile number or person. Several operators have begun looking at taking advantage of this to provide single sign-on services.

However, Facebook also offers an authentication service called Facebook Connect, which allows end users to login to third party websites with their Facebook credentials, creating a single sign-on user experience. According to LeadLedger around 15 % of the top Fortune 500 web sites support this integration, with up to 25 % of websites in specific categories as of June 2013 [18]. More than 10 million applications and websites have been integrated with

Facebook as of the end of 2012 [19]. This can be seen either as a challenge or opportunity depending on the network operators business model and their desire to integrate with Facebook.

1.2.2 Content creation

More and more end users are being responsible for the content available on the internet. YouTube, a website that allows end users to upload their videos, receives more than 400 hours of video content uploaded every minute in 2015, rising from 100 hours of video content per minute in 2013 [2, 20]. SoundCloud, a website that allows end users to upload their own sound recordings, sees around 11 hours of audio uploaded every minute as of May 2013 [2]. The most popular social platforms see more than 500 million photos uploaded every day [2]. This huge increase in content coming from the end user poses challenges in both the technical and business areas for network operators. Bandwidth requirements have swelled considerably, and end users are demanding more unique, interactive service experiences. They are no longer content to have content parcelled out to them by large content providers at what is seen as a high cost compared to the free or advertising sponsored services available on the internet. New network technologies are needed to cope with the increased demand for data and richer services.

1.2.3 Google Voice

A relevant example of a service being designed around the end users' needs is Google Voice. Google Voice is an example of a non telecommunication operator beginning to control the relationship of the end user. It offers advanced features such as voicemail, call history, conference calling, call screening and blocking, transcription of voicemail etc. This leads to people associating positive aspects with the service provider, and the network provider becomes a straight cost instead of adding value. The assumption here is that the

end user will migrate to the lowest cost network provider as the end user experience is provided by another service provider. The difference between network provider A and network provider B becomes less obvious if they can use Google Voice with the same number on both of them and get the same experience. With more than 10 million installations via the Google Play store it is a sign of the potential disruption that is possible [21].

1.3 Next Generation Networks

The relevant standards bodies and telecommunication operators have not been idle while the Internet revolution was taking place. The traditional revenue streams such as voice calls or messaging are drying up under competition from over the top providers such as Skype or WeChat. Different network operators have handled this in different ways and there are several options available. The network operator can take advantage of the fact that their network can provide a better quality of service versus the best-effort nature of the Internet. They can exert control over their own infrastructure and tightly control all aspects of the end users' experience. Another option available is to allow third parties to make use of their own network infrastructure to provide customised, enhanced services. A third option is to enlarge their presence in the content space and receive a cut of any revenue generated by their own users (e.g. offering specials to their end users for a percentage of the resulting sales).

In order to remain competitive and avoid becoming a simple bit pipe operator providing access networks, operators need to reduce costs and increase their ability to offer new and innovative services. The old model of developing completely isolated point solutions for each service is no longer sustainable. This has led the move towards a single converged, all IP based packet switched network, which is often referred to as a Next Generation Network.

1.3.1 Principles

Traditional telecommunication networks have been circuit based networks. These networks establish a particular path for information to flow along and reserve the path for the duration of the session, regardless of whether or not any information is being sent at that particular time. This is both inefficient and costly, as the resources are considered in use even when not transferring information. Voice communication only utilises 40 % of the reserved resources in a circuit switched session. For general purpose data transmission, which is characterised as having more bursts of transmission followed by idle time, this figure is even lower [22].

In contrast to the circuit switched network mentioned above, packet switched networks are much more efficient at utilising resources. These networks break down the information transmitted into chunks called packets, which are then routed over a shared network. This is the preferred network structure for transmitting data, and is normally how data services are provided to the end user.

In addition to the inefficient circuit switched networks, network operators have mainly deployed specific point solutions. This is a service which has been developed from the ground up to offer specific functionality. Common functionality (such as authentication or authorisation) is re-implemented for every specific service. It is not possible to reuse pre-existing work, increasing the cost and time needed to develop every new service. Telecommunication operators have provided high availability, carrier grade services and have focused on person to person communication with independent vertical solutions. Generally the network operators have not been very open to third party developers offering their own services over the network operators infrastructure.

Next Generation Networks (NGNs) aim to replace the traditional circuit switched network as well as the packet switched data network with a single, unified packet switching network. This is more efficient, both in resource util-

isation as well as initial capital investment and ongoing maintenance costs, as two separate networks do not have to be built and maintained. This is one of the key features of NGNs, but there are several distinct features as standardised by the International Telecommunications Union (ITU) [23]. Some of these are extracted and listed below:

- packet-based transfer of information
- separation of control signalling from transport mechanisms
- decoupling of service functionality from transport functionality
- support for a wide range of services based on service building blocks
- real-time and non real-time services
- end-to-end Quality of Service (QoS)
- third party service providers
- independence of service-related functions from underlying transport technologies
- support of multiple last mile technologies

As can be seen by the list above the aims of a NGN is not simply to move to a packet based network, although that is a large factor. NGNs also aim to increase the number of services available to the end user, as well as the flexibility of these services. They are designed to allow the rapid prototyping and development of new services by a much wider selection of developers.

The introduction of service building blocks means that rapid service creation should be possible by allowing the reuse of different components. Each service is able to take advantage of common functionality such as the various Authentication, Authorisation and Accounting (AAA) subsystems provided by the network.

1.4 The IP Multimedia Subsystem

One of the most promising NGNs is the IP Multimedia Subsystem (IMS). The IMS was originally designed to evolve the cellular networks, taking advantage of experiences learnt through the development of the Internet and the previous traditional telecommunication frameworks. The scope was later expanded to cover fixed line networks as well. The IMS is standardised by the Third Generation Partnership Project (3GPP) and has gone through several different releases, with each release offering new functionality or changes to existing functionality [24]. The aim of the IMS is to provide a NGN that can be used to offer rich multimedia services, allowing the network operator to offer compelling services over their infrastructure on a seamless IP based network [25]. This packet based network would form the core part of a telecommunication network. The IMS does not deliver a set of client facing services, but rather acts as the control platform for allowing new services to be developed and operated. This allows them to move away from being access providers and allows them to investigate different revenue streams to replace the ones that have come under pressure from over the top providers as discussed above. These rich multimedia services are provisioned through the use of Application Servers (ASs) that are developed by either the network provider or a third party. Multiple services can be invoked during one session, with each AS providing a different set of functionality [26]. There is a large drive to open up service creation to a wider target audience, allowing many more parties to take part in service creation [27].

While it took a while for IMS to gain momentum, 26% of operators were planning to have deployed IMS based networks, with another 29% in extended field trials or running limited commercial services by 2012 [4]. Since then IMS deployments have increased dramatically, in part bolstered by the investment in Voice over LTE (VoLTE) which can take advantage of the Quality of Service (QoS) offered by the IMS. More than 118 operators in 56 countries have invested in VoLTE deployments, studies or trials by 2015 [28].

1.4.1 Services

The IMS does not standardise a set of services to be offered as the idea is to provide a platform that will allow a vast number of different services to be developed which may or may not be feasible at the moment. The intention is to allow the network operators and third party suppliers (including Internet based application providers) to take advantage of the functionality exposed by the IMS to develop innovative services [29]. While the 3GPP does not standardise the IMS based services, the Open Mobile Alliance standardises several building blocks which can be used as components when designing new services. One example of a standardised building block is the Presence subsystem [30]. This subsystem provides the necessary components to indicate the availability, or willingness to communicate, of an end user. It can also include details as to the types of communication the end user is able to receive. This allows services to be developed that can be contextually aware of the end user and change their functionality to suit the end user's current needs. There are similar enablers standardised for other components such as location services, instant messaging etc. Services are provisioned through the use of Application Servers (ASs) which are either developed by the network operator or a third party. These ASs offer a variety of services, and more than one AS can be used per session depending on the functionality offered.

1.4.2 Signalling

The IMS uses the Session Initiation Protocol (SIP) [31] as the main control signalling in the IMS [32]. SIP is used for the establishment, modification and termination of multimedia sessions. The SIP messages are sent over either the User Datagram Protocol (UDP) [33] or the Transmission Control Protocol (TCP) [34] for establishing the session, and the actual multimedia is handled by a different protocol, e.g. Real-time Transport Protocol (RTP) [35]. This is paired with the Diameter protocol [36] which is used for Authentication, Authorisation and Accounting [37]. These protocols are widely deployed on

the Internet, and are fairly well understood by a wide range of developers. Thus less specialised knowledge is needed to develop telecommunication services for the IMS compared to previous telecommunication networks such as Intelligent Networks. It should also decrease the difficulty in porting existing SIP based voice applications from the Internet to the IMS platform. The use of these protocols follows the principles established for NGNs by separating out the control signalling from the lower layers, decreasing the difficulty in developing services as well as opening the service creation process to more developers.

1.5 Service Orchestration

Service orchestration refers to the overall orchestration or organisation of individual services. This includes their activation or execution as well as the surrounding mechanisms needed to achieve control over the overall session [38–40]. While the standard approach to service triggering is well defined by the 3GPP [26, 41], a more flexible approach with added intelligence is required.

1.5.1 Feature Interaction

The IMS is intended to increase the number of services available to end users as well as enriching these services with new capabilities. It is envisaged that many more complex services will form part of the normal communication package available to the end user. These individual services can be combined either by themselves or with other services to form a new complex service that is seamlessly offered to the end user as a single enriched service. This increase in services and service complexity has exacerbated the traditional feature interaction problem found in previous telephony network architectures [42]. The feature interaction problem can be broadly described as the

situation where services or *features* run fine in isolation, but present problematic results when run simultaneously with other services. Feature interaction problems can be solved either proactively or reactively. Proactive mechanisms prevent any conflicting services from being activated together, where reactive mechanisms work by disabling one or more services when a conflict is detected. Services that are expected to pose challenges in this area include the well known call control services such as *Call Barring*, *Call Forwarding Unconditional*, or *Call Screening* services (both *Originating* and *Terminating*) as well as *Caller Identification* services (e.g. *Calling Line Identification Restriction*). An example can be seen in the following scenario[43]:

1. User Carol has a configured *Call Forwarding* service which forwards all incoming calls to user Eve.
2. User Bob has a configured *Originating Call Screening* service which prevents outbound calls to Eve.
3. User Bob calls user Carol.

The goals of the above services are contradictory. Either the *Call Forwarding* service will succeed, resulting in the *Originating Call Screening* service being bypassed, or the *Originating Call Screening* will take place, bypassing the configured *Call Forwarding* service. This and similar situations lead to undefined behaviour within the system, resulting in a poor user experience as well as system instability. Also to be considered are auxiliary systems such as the charging functions. In the scenario given above, it is possible for the end user to be charged for both services although only one service was executed successfully. Currently there are no standardised service conflict detection and resolution mechanisms in the IMS. As it is envisaged that the IMS will support an ever increasing array of services, offered by many different parties over the same network, service conflict detection and resolution mechanisms will become more important as the number of available services increases.

Systems used to solve these problems are generally classified as either *offline* or *online* conflict detection and resolution systems [44–46]. Methods where

conflicts between services are detected before service invocation are defined as offline methods, whereas conflict detection that takes place in real-time during service invocation is defined as online conflict detection. Due to the nature of services and the large number of possible service combinations it is thought to be infeasible to have a purely offline conflict detection based system [43]. Furthermore the conflict detection and resolution methods can be considered either *static* or *dynamic*. Static conflict detection and resolution mechanisms rely on fixed, predictable parameters to prevent conflicts and dynamic resolution procedures take place in real-time based on dynamic, changing information.

1.5.2 Contextual Service Triggering

Services are controlled in the IMS by what is known as the Application Triggering Architecture (ATA) [26]. This architecture controls how services are invoked, with the Serving Call Session Control Function (S-CSCF) determining which Application Server (AS) will be invoked for any particular initial request. It does this by comparing the request against a list of filters to determine where to send the request. These filters are statically set by the network operator and base the service triggering or service request routing only on certain parameters such as the URI in the “TO” header of the request or the type of SIP message. It is not possible to activate services depending on the results of a previous service as this routing takes place only on the initial request. Contextual information (such as the status of the end user) is not taken into account when making these decisions. Contextual information is seen as information from internal or external sources that provide relevancy to the user’s current actions or status. This information can include the status of the end user (e.g. busy or offline), the location of the user (e.g. at work or at home) as well as preferences or capabilities for different types of communication (e.g. voice or video call functionality). Currently the negotiation of device capabilities is handled on a per session basis between two end points using SIP and the Session Description Protocol (SDP) [47].

A simple example of contextual service triggering would be to redirect incoming voice calls from work related contacts to a voicemail service if it was currently outside business hours and the end users location was not set to their work location. A service like this can only be currently offered by providing a custom designed Application Server (AS) to fulfil this exact service.

1.5.3 Service registration

The current literature details the increase in the type and number of services being developed or offered to the end user [4, 27]. However, much of the focus has been on exposing APIs or network functionality to third parties [48]. Traditionally the addition of any service is handled in a very manual fashion by the network operator, working on a per Application Server (AS) or service developed basis. It is doubtful whether this approach would be scalable should the number of services increase as expected.

1.5.4 User centric service control

At the moment the IMS has fallen behind in embracing the concept of user centric computing. Which services are triggered for which requests is still largely set by the network operator in a static fashion. The end user does not have any control of how or when their particular telecommunication services are activated for specific messages.

1.5.5 Service Triggering in the IMS

The component introduced by the 3GPP to improve the service triggering architecture was known as a Service Capability Interaction Manager (SCIM). It was standardised by the 3GPP as a black box, without any details on its

inner workings [49, 50]. It was proposed that the SCIM be a stand alone component between the S-CSCF and the IMS AS. The interface between the SCIM and the S-CSCF as well as the interface between the SCIM and the AS was based upon SIP. Separating the SCIM out into the application layer makes sense as it needs to perform unique duties between the different SIP ASs as well as utilise non uniform data from various unspecified sources.

In parallel a standardisation process was underway to define a Service Broker (SB) for the Open Service Access (OSA) / Parlay context under the 3GPP Release 7 specification. This work item's purpose was to control service interactions between OSA services. The desire to allow service interactions between OSA applications as well as non-OSA applications was raised, but left for further study [50, 51].

To examine both these options and decide on the way forward the 3GPP commissioned a study called “*On Architecture Impacts of Service Brokering*”, which concluded near the end of 2008 [46]. This study is discussed in depth in Chapter 2, however it should suffice to state that the objective of the study was “to study if there is enhancement needed to the current service interaction management architecture (e.g. SCIM as part of AS and Service Broker as part of OSA Service Capability Server)” [46]. The study concluded that improvements were needed in this area, and recommended further work. Due to the similar nature of the functionality between the SCIM and the SB, as well as the fact that the 3GPP is now using the term Service Brokering to discuss what was previously known as Service Capability Interaction Management the two terms will be used interchangeably in this thesis.

1.6 Thesis Objectives

While the IMS has made good strides towards providing end users with a platform that will enable a wide variety of multimedia enhanced services, more research is needed around the provisioning of services. As discussed

above, the 3GPP study into issues revolving around the service triggering architecture inherent in the normal IMS architecture found several areas requiring further work [46]. These findings are investigated in more depth in the next chapter, but mainly revolve around the inability of the IMS to make service triggering decisions based on external or contextual information according to the end users desires.

This thesis aims to solve the problems discussed above through the use of several distinct objectives.

It is necessary to conduct a full review of the standardised architecture for service triggering in the IMS together with the current state of art that aims to solve this well-established problem. While this problem has been identified in numerous different telecommunication networks before [42, 45, 52], the IMS poses several distinct challenges due to the existing architecture and depth of the signalling paths. As any future enhancements need to be backwards compatible with the existing IMS networks and standardisation process, a full overview of the current architecture needs to be conducted to identify non invasive modifications.

There is a need to establish requirements or constraints for any improvements to ensure that the work completed is relevant to the existing architecture and solves the problems identified by the 3GPP in a feasible manner. As the service triggering architecture forms part of the network operator's core network and is expected to require large capital expenditure, it is important to have highly specific, well defined requirements to justify these resource allocations.

Furthermore improvements to the existing IMS architecture need to be identified, taking into account both the standardised architecture, the requirements for improvements as well as related work in this area.

The primary objective of this thesis is to design and implement an enhanced, standards compliant service triggering architecture. It must unify the identified improvements and allow the end user to modify his or her preferred

order of service execution. Contextual information needs to be taken into account, allowing different services to be triggered depending on the current context of the user. The functionality provided by the architecture can be broken down into three main areas.

1.6.1 Dynamic Service Orchestration

The main area of proposed functionality is the dynamic routing of SIP messages based on contextual information. A specification for contextual information and routing decisions is needed, as well as the creation of the necessary execution environment so that the results from a particular interaction with one AS can be stored and used as inputs to the next interaction with the same or different AS.

1.6.2 Service Registration and Monitoring

To make informed decisions about service triggering and to avoid service conflict the proposed architecture will need to have certain information about each offered service. A manual compilation of this information has a high degree of management overhead and is not consistent with the goals of the IMS. The IMS intends to introduce many different services provided by many different service providers, and as such a manual composition of service information will not be suitable as it does not scale. A way for each Application Server to dynamically inform the architecture about its capabilities and current status is needed. A secondary goal is to use this information to detect and avoid potential service conflicts.

1.6.3 User Policy Creation and Modification

The final functionality needed is the exposure of the new architecture to the end-user. The user must be able to tailor their service usage to create a better user experience.

1.7 Thesis Scope and Limitations

The scope of this project is limited to the design, development and proof of concept implementation of the proposed service triggering architecture. This architecture forms part of the core IMS network, and as such the access networks will not be examined. Central to the project are issues relating to the control of SIP signalling which is internal to the core network as well as the control signalling between the end user and the network. Prototype services will be developed to serve as mock examples of services to be used in the architecture. Complex multimedia handling will be avoided where necessary as the main focus of the thesis is the establishment of the session and the corresponding control signalling. To fulfil the goals of this thesis not only is it necessary to develop the core network architecture but also the necessary end user client software that can interact with the architecture. While service conflict detection and resolution mechanisms are introduced as part of the framework, the algorithms used for these mechanisms are limited to rudimentary proof-of-concepts to determine the feasibility of the overall architecture, and are left as areas of future work. Where possible the proposed architecture will adhere to the relevant 3GPP standards, but as the project proposes new functionality, some extensions to the architecture and protocols may prove necessary. These extensions must be limited to non-invasive changes that are backwards compatible with existing IMS architecture as there are already pre-existing IMS networks.

While the IMS is intended to serve as the core part of a network operators network, it is envisaged that interfaces with legacy services or applications

will need to be developed through the use of interoperability gateways. The recommended deployment of new services for the IMS is through SIP Application Servers, and this is the model followed for the services discussed in this thesis. Previous service development models are considered out of scope of this work as well as the integration of the IMS with future network evolutions such as the Evolved Packet Core. The IMS network remains a core part of the System Architecture Evolution (SAE) which will be used for the Long Term Evolution (LTE) networks currently being developed, and hence issues around service triggering in the IMS core are still relevant.

1.8 Contributions

There are several limitations with the current standardised mechanisms for service invocation in the IMS. These have been noted by the 3GPP [46]. This work enhances the existing IMS session setup call flow to tackle these limitations. It does this by providing a new architecture that involves the addition of new core elements, but reuses existing protocols and conforms to the specifications in order to remain backwards compatible. This allows the introduction of this system to pre-existing IMS based networks. It introduces customisation by the end user while offering a different perspective on the service conflict detection and session setup delay problems in comparison to the existing work. The author believes that the proposed system has advantages over current existing work in this area as detailed below while providing a viable solution to the problems highlighted by the 3GPP work group. The major contributions are listed below:

- Critical review of the existing standards and literature in this area and defined requirements for the identified areas which require further work.
- Design, implementation and evaluation of a system to allow automatic service registration and monitoring in the IMS.

- Design, implementation and evaluation of a new architecture that allows dynamic service orchestration in the IMS. This involves the development of several different prototypes, including a Service Capability Interaction Manager to perform the dynamic service routing as required by the thesis objectives, a User Policy Repository Server to allow the end user to create and store service triggering scenarios, a Service Repository Server to store service information and a Context Server to aggregate the end users contextual information. All these components have been released as Open Source Software to aid further research in this area.

In addition several smaller prototypes were developed. These include:

- An open source, prototype IMS client for the windows operating system. It is capable of registering with the IMS core network, performing voice and video calls, instant messaging, presence updates and notifications as well as utilising a network stored address book for contact information.
- An Open Source SIP library which can be used to create both IMS based or non IMS based VoIP applications.
- Various Application Servers that can be used to develop “proof of concept” services.

Several papers have been published during the course of this research project:

- D. Waiting, R. Good, **R. Spiers**, and N. Ventura, “Open source development tools for IMS research”, *Proceedings of 4th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM'08)*, March 2008. ISBN 978-963-9799-24-0.

- **R. Spiers** and N. Ventura, “An evaluation of architectures for IMS based video conferencing”, *Proceedings of the Southern African Telecommunication Networks and Applications Conference (SATNAC 2008)*, September 2008. ISBN 978-0-620-41696-2.
- D. Waiting, R. Good, **R. Spiers**, and N. Ventura, “The UCT IMS client”, *Proceedings of 5th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM’09)*, April 2009. ISBN 978-1-4244-2846-5.
- **R. Spiers** and N. Ventura, “Dynamic service orchestration in the IP Multimedia Subsystem”, *Proceedings of the Southern African Telecommunication Networks and Applications Conference (SATNAC 2009)*, August 2009. ISBN 9780620441070.
- **R. Spiers**, R. Marston, R. Good, and N. Ventura, “The UCT IMS IPTV initiative”, *Proceedings of the 2009 Third International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST ’09)*, September 2009. ISBN 978-0-7695-3786-3.
- **R. Spiers** and N. Ventura, “A Converged IMS Client for the IP Multimedia Subsystem”, *Proceedings of the Southern African Telecommunication Networks and Applications Conference (SATNAC 2010)*, September 2010. ISBN 9780620479349.
- **R. Spiers**, R. Good, and N. Ventura, “Service Orchestration in the IP Multimedia Subsystem”, *Service Delivery Platforms: Developing and Deploying Converged Multimedia Services*. Auerbach Publications, March. 2011. ISBN 978-1-4398-0089-8.
- **R. Spiers** and N. Ventura, “User customisation of service request routing for the IP Multimedia Subsystem”, *Proceedings of the Southern African Telecommunication Networks and Applications Conference (SATNAC 2011)*, August 2011. ISBN 978-0-620-50893-3.

- **R. Spiers**, N. Ventura, and T. Magedanz, “Improvements on the application triggering architecture for the IP Multimedia Subsystem”, (AFRICON 2011). September 2011. ISBN 978-1-61284-992-8.
- J. Mwangama, **R. Spiers**, N. Ventura, and T. Magedanz, “Past and current IMS testbed initiatives: The UCT IMS and EPC testbed”, 2012 IEEE Globecom Workshops, December 2012. ISBN 978-1-4673-4942-0.

1.9 Thesis Outline

The remainder of this thesis is structured as follows. Chapter 2 contains an overview of the existing standardised IMS architecture, with a particular focus on the Application Triggering Architecture. It contains a discussion of the relevant results from the 3GPP study identifying the need for further work in this area. This is followed by a review of the current related work in Chapter 3, including both functional improvements as well as proposed improvements to the actual architecture. Chapter 4 introduces the design of the solution architecture proposed in this thesis. It introduces the new elements proposed as well as the reasoning behind their formation, together with the lessons learnt from the related literature. A prototype implementation of the proposed architecture is discussed in Chapter 5. This implementation was tested using a standards compliant test-bed which is detailed in Chapter 6. Both functional as well as performance test results from this implementation are discussed in Chapter 7. These results demonstrate that the architecture works as designed, and highlights the feasibility of the introduced architecture. The thesis is concluded in Chapter 8 with an overview of the work performed together with recommendations for further research in this area.

Chapter 2

Application Triggering Architecture in the IMS

The 3GPP applied lessons learnt through the development of the Internet during the standardisation of the IMS. A heavy focus was placed on using open, widespread protocols such as the Session Initiation Protocol (SIP) [31] to ease interoperability with existing services as well as encourage uptake by service developers [26]. The resulting architecture and control signalling arising from the standardisation process of the IMS is detailed below. This is followed by a discussion around the 3GPP study which was commissioned to determine if the architecture met all the high level requirements originally envisioned for the IMS. This study was a large motivating factor for this dissertation as it found deficiencies as detailed in the closing sections of this chapter.

2.1 Current Functionality

As per Next Generation Network (NGN) guidelines the control signalling is separate from the media flow or session content [23]. The control signalling

is based on SIP, while the media content may use several different protocols. The routing of service requests or control signalling within the IMS uses well recognised SIP Uniform Resource Identifiers (URIs) or non SIP Absolute URIs. These indicate the destination for different SIP requests. The alternative Absolute URIs were defined later than the initial SIP release, but are well standardised by now in RFC 3986 [53]. While the majority of the following work deals with the control signalling for session establishment, and hence SIP, the IMS also utilises the Diameter protocol [36] to transmit sensitive information such as authorisation or charging related information.

2.1.1 Architecture

The main IMS components relevant to this thesis are the Proxy Call Session Control Function (P-CSCF), the Interrogating Call Session Control Function (I-CSCF), the Serving Call Session Control Function (S-CSCF), the Home Subscriber Server (HSS) and the various Application Servers (ASs) that provide the actual services. These elements will be detailed in the coming sections. The end user's client equipment or software is also important to the work contained in this thesis as it revolves around the end user's use and control of services. An example of the standardised IMS architecture can be seen in Figure 2.1. This excludes the majority of interfaces between different elements as well as the various media handling components and interoperability gateways as they are out of scope of the work presented. The end user terminal is also shown interacting with an XML Document Management Server (XDMS). The XDMS is viewed by the Open Mobile Alliance (OMA) as a useful enabler to manage groups, contact lists etc [54].

The P-CSCF is the first point of contact towards the core network from the end user. The location of the P-CSCF is discovered through means outside the scope of this document, such as during the address assignment which happens during network attachment. Once the User Equipment (UE) has determined the location of the P-CSCF it serves as an entry point for further

communication. The P-CSCF acts as a SIP Proxy (as defined in RFC 32161 [31]), forwarding all messages towards the core network on behalf of the end user. The P-CSCF performs several different functions including:

- Forwarding the SIP register request received from the UE to the appropriate entity based on the home domain name.
- Forwarding appropriate SIP requests to the S-CSCF as determined through the registration procedure.
- Forwarding the responses to the UE.
- Performing SIP message compression / decompression.

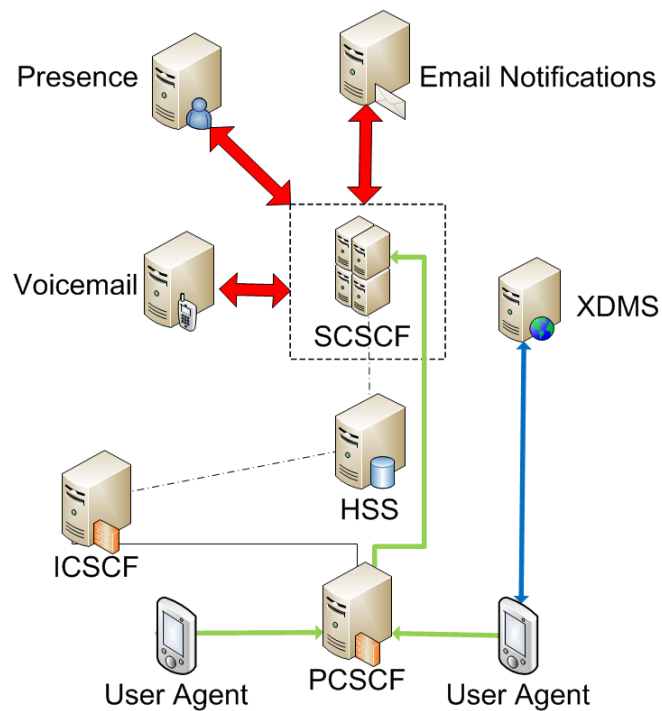


Figure 2.1: Standardised IMS architecture for service triggering.

The I-CSCF is the contact point for all connections destined to a user of that network. This includes both local and roaming users currently attached to

the network. The I-CSCF is responsible for forwarding on messages received from another network towards the S-CSCF. It can query the HSS and if it determines that the destination of the session request is not within the local IMS it may forward the message on or reject it with the appropriate failure code.

Once the end user is registered on the network, it is assigned a S-CSCF to handle the control signalling for the end user's call setup and other services. The S-CSCF is where most of the SIP processing for session control is handled. It maintains the required level of service state for the IMS to function, and may act as a SIP Registrar. It also performs session control for the end user's sessions, rejecting IMS communication from/to barred Public User Identities. One of its main functions is to act as a SIP Proxy, forwarding on SIP requests for further processing. The SIP signalling that passes through the S-CSCF can be evaluated by the S-CSCF to ensure that it conforms to the IMS communication service definition as indicated by the end users subscription. More specifically, the *S-CSCF is the entity responsible for determining the signalling path of a SIP Session*. It determines the next hop for the SIP messages received from either an AS or the P-CSCF, triggering new services or returning responses to the end user via the P-CSCF. The S-CSCF also plays roles in various other areas, such as facilitating the communication of policy information between the HSS, the P-CSCF and/or the UE. However the other roles are not relevant to this thesis and have been omitted. Further information about these roles can be found in the 3GPP Technical Specification 23.228 [29].

The HSS acts as a repository for the end user information. It is the master database that contains the end users subscription information necessary to support the network entities that actually handle the end user's calls or sessions. This information contains:

- Security information used for authenticating the end user as well as authorising his or her actions

- The user identification and addressing details
- User profile information

The HSS functionality can also be considered a superset of both the traditional Home Location Register (HLR) as well as the Authentication Centre (AUC). These functions are outside the scope of this thesis, and will not be discussed further. Information about the full scope of the HSS can be found in 3GPP Technical Specification 23.002 [55].

The ASs offering value added IMS services can reside either in the user’s home network or in a third party location. This third party could be a stand alone AS or a more complicated network [29]. These SIP ASs host and execute the actual service functionality, and influences the SIP session through the use of the IP Multimedia Subsystem Service Control (ISC) interface with the S-CSCF. The ISC interface also allows the S-CSCF to notify the AS of various events, such as the implicit registered Public User Identities and / or the registration state and capabilities of a particular UE. Whether or not an AS is invoked is controlled by the S-CSCF based upon information received from the HSS. This information is stored and communicated to the S-CSCF on a per AS basis. In the current IMS standards, the S-CSCF does not perform any further service management, including handling service interaction [29].

2.1.2 Service Triggering

As stated above, the main component for session control is the S-CSCF. The S-CSCF is the core element that decides how incoming SIP requests are routed. The P-CSCF forwards the SIP request received from the end user to the S-CSCF. The S-CSCF then consults a list of filters. These filters are known as the initial Filter Criteria (iFC) and are downloaded from the HSS when the end user registers on the network. The iFC contain a set of headers that can be matched against the SIP request (e.g. matching against values stored in the “TO”, “FROM” headers etc), a priority field as well as details

of the destination AS onto which the S-CSCF will forward the request. The priority field is used to determine the order followed if more than one iFC matches a particular message. If more than one AS matches the SIP request according to the different iFC the SIP request will be forwarded to the AS indicated by the iFC with the highest priority. It will then be forwarded to the next matching AS if the initial AS did not terminate the SIP session. Due to this behaviour there is a direct correlation between the session setup delay and the number of AS involved in the session.

Service Point Triggers

Service Point Triggers (SPTs) are defined as items in a SIP request that can be identified with the possibility of setting Filter Criteria (FC) on them, and thus are values that can be used to trigger services. The following SPTs are defined in 3GPP TS 23.218 [26]:

- The initial SIP method
- Registration type (e.g. re-registration, de-registration etc)
- The presence or absence of any particular SIP header
- The value of any particular SIP header
- The value of the Request-URI
- Session Description Information
- Direction of initial request (i.e. from the end user or towards the end user)

Filter Criteria

When an incoming request matches a specific SPT, the request is forward to the particular AS that has been configured to receive the message according

to the Filter Criteria (FC). These FC are stored on a per user basis, and are stored in the HSS in what is known as the Application Server Subscription Information. These stored FCs are referred to as the initial Filter Criteria (iFC). Originally the concept of subsequent Filter Criteria (sFC), introduced in 3GPP TS 23.218 version 0.6.0 [49], leads to the need to include “initial” as part of the FC name. Subsequent FC revolved around the AS signalling to the S-CSCF to allow for dynamic definition of SPTs during service execution. However, sFC were dropped from later standards as it conflicted with the ideas behind the ISC interface, namely that it was based on SIP and the sFC would break the SIP standard. The 3GPP decided to retain the iFC term going forward although the sFC concept is no longer used. This information is communicated to the S-CSCF by the HSS via the Cx interface during the end user’s registration. The HSS can also send updated information to the S-CSCF later on as specified in 3GPP TS 29.228 [56].

It is possible that more than one SPT (with possibly more than one destination AS) may match a particular SIP request. In this case, the S-CSCF forwards the request to each AS sequentially, using the response from the first AS as the input to the second AS. The order in which each AS is placed is determined by the priority field contained within the Filter Criteria. Should a particular AS not respond to a particular request, the S-CSCF shall continue handling the session according to rules which are present on the SPT. These rules indicate to either continue handling the session with the remaining matching AS or to abandon the session entirely. The following information is standardised as part of the iFC in 3GPP TS 23.218 [26]:

- Address of the Application Server
- Priority of the Filter Criteria
- One or more SPTs, linked by means of a logical expression (AND, OR, NOT etc)
- Default handling rules (terminate or continue as described above)

Request processing

The S-CSCF requests the list of iFC for a served user from the HSS upon user registration. This procedure does not have to be repeated should the S-CSCF determine that it currently has a valid set of iFC (i.e. from a previous request). When the S-CSCF receives a request on the *Mw* interface it evaluates the iFC in a sequential manner. The *Mw* interface is based on SIP. Each SPT from each iFC is compared against the initial SIP request in a specific order as determined by the priority field contained in the iFC. SIP REGISTER requests as well as events that cause network-initiated de-registration are handled slightly different from normal SIP requests, as third-party REGISTER requests are sent to each AS that matches the iFC sent from the HSS for the REGISTER request. On receipt of any other SIP request the S-CSCF applies the following procedure, which is detailed in TS 23.218 [26]:

1. Order the iFC into a list based on their priority field. This list may not change until the processing of the request is finished.
2. Parse the message into its various components that may be matched against SPTs.
3. Match the trigger points from the next highest priority iFC against the SPT from the request.
 - (a) If it does not match the S-CSCF moves to step 4
 - (b) If it does match the S-CSCF:
 - i. adds an Original Dialogue Identifier (ODI) which allows the S-CSCF to recognise the message again even if the dialogue identification has changed e.g. due to third party call control being applied by a AS
 - ii. forwards the request to the particular AS indicated by the iFC

- iii. proceeds with step 4 if a request with the same ODI is received from the AS
4. Repeats step 2 and 3 for every filter criteria which was originally included in the list arranged in step 1.
5. Route the message according to default SIP routing rules.

If the AS returns a SIP final response for that request, the S-CSCF abandons further matching of lower priority triggers as the final response indicates that the AS terminates or controls the session. Any AS that has been triggered is free to remain in the signalling path by adding its own address to the “RECORD-ROUTE” header. If the AS does not add itself to this header, further SIP signalling in the session will not be passed through the AS.

An example of an SIP INVITE request being served by an AS can be seen in Figure 2.2. The end user generates the request on their UE which forwards the request to the P-CSCF. The P-CSCF then forwards the SIP request received from the end user to the S-CSCF. The S-CSCF consults the list of iFC previously downloaded from the HSS when the end user registered on the network, and directs the SIP request to the appropriate AS. The AS accepts the session and sends a SIP response back to the S-CSCF which is forwarded to the UE via the P-CSCF.

As discussed above more than one AS can be involved per SIP session. For example, an end user may have a call screening service which rejects callers on a black list, followed by an application server that acts as a back-to-back user agent (B2BUA) which establishes the actual call to the end user. The session request would first be sent to the call screening service, and only pass on to the next application server if allowed. The ordering of the messages for such an example of service chaining can be seen in Figure 2.3. It is important to note that there is a direct correlation between the session setup delay and the number of AS involved in the session as well as the number of messages generated. Further SIP messages for this session do not have to flow through

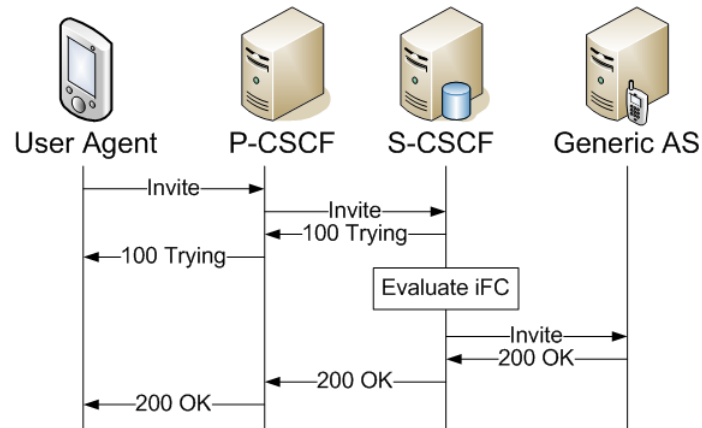


Figure 2.2: Evaluation of iFC in the IMS.

all ASs as it is up to the AS to decide whether or not they wish to be included in the signalling path by modifying the “RECORD-ROUTE” header.

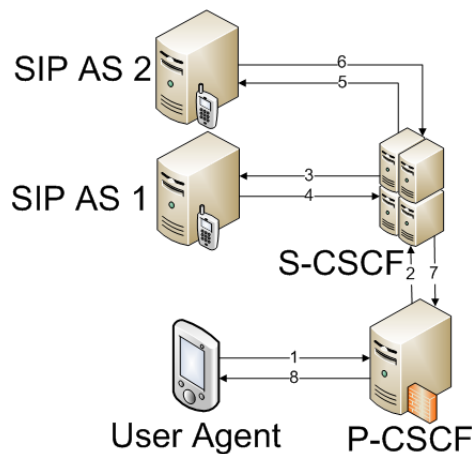


Figure 2.3: Service Chaining in the IMS.

2.1.3 Discussion

While the standardised architecture has been proven viable by real world IMS implementations, it has attracted a fair amount of research as to whether or not it is as powerful and flexible as required by the envisaged set of NGN services that will be developed in the future. Of particular note is the fact

that the 3GPP, which is the standardising body responsible for the IMS, has conducted a study to determine if the currently standardised architecture is suitable for fulfilling the identified goals of the IMS platform. This study is discussed in the following section.

2.2 Impacts of Service Brokering - 3GPP Study

The previous section has introduced the Application Triggering Architecture as currently specified by the 3GPP in TS 23.218 [26]. However, the actual high level requirements for IMS services are standardised by the 3GPP in Technical Specification 22.228 [41]. These are general requirements for IMS based services including requirements such as utilising Public User Identities for user identification, or providing the same service across multiple access networks etc. During the standardisation process the 3GPP established a study, TR 23.810, to determine whether the current Application Triggering Architecture was flexible enough to satisfy these requirements [46]. This study was entitled “*On Architecture Impacts of Service Brokering*” and was concluded at the end of 2008. It found several areas requiring further work. Specifically, the study focused on whether or not there was sufficient interaction management between ASs as well as determining if there was a need for further enhancement of the architecture and protocols currently in use. The study defined several different requirements, scenarios and the general language to describe various functions related to service interaction management.

2.2.1 Motivation for improvements

A specific use case or *scenario* is specified by the 3GPP as motivation for improving the service triggering architecture in the IMS [46]. This scenario includes the following four services, each provided by a separate AS:

1. Incoming call logging (ICL)
2. Incoming call barring (ICB)
3. Email notification (EN)
4. Denied termination (DT) based on presence

The desired service activation scenario is as follows:

- If the end user's presence or status is currently set to indicate that they are not available, e.g. a value of "*Do not disturb*", the IMS will not allow any incoming requests (utilising the DT service), but instead record the information of the requester (utilising the ICL service) and send an email notification to alert the user (using the EN service).
- If the end user's presence or status is currently set to indicate that they are available, e.g. a value of "*Online*" or "*Available*" all incoming requests will be passed to the ICB service for screening against a black list.
 - If the caller is found to be present in the black list of the ICB service, the request will be passed to the ICL service for logging and then to the EN service to alert the end user.
 - If the caller is not barred by the ICB service, i.e. they are not in the black list, the request will be routed normally i.e. according to SIP routing rules.

This scenario is not currently possible with the standardised architecture, as there are several different service combinations required. A session could resolve in at least three different ways. For example:

1. DT followed by ICL followed by EN
2. DT followed by ICB followed by ICL followed by EN

3. DT followed by ICB

Which of the above service combinations is correct for any given service requires external information and requires the history of previous interactions (e.g. the result from the DT service) to be used to determine future interactions. This is not possible with the current functionality offered by the application triggering architecture.

2.2.2 Definitions

Throughout this thesis the phrases “*service triggering*” or “*service invocation*” are used interchangeably. In addition, *service interaction management* or *service brokering* are similarly used. The definitions described below are specifically taken from the 3GPP study to align the rest of this thesis with the terms and phrases used by the standards body responsible for the IMS. There are small discrepancies between terms used in the related work so to avoid confusion the specifications from the 3GPP are taken as the canonical definitions. They are listed below as they will be used to categorise both the related work in this area as well as the architecture proposed in this thesis.

Online interaction management

Online interaction management is simply defined as the functions required and steps taken to handle any potential feature interaction during a live session, i.e. the steps are taken either during the establishment of an end user’s session or during the session itself. The definition is purely about the time at which management features are executed. The standardised iFC are considered to be a form of online interaction management.

Offline interaction management

Offline interaction management are actions that are taken before the end user establishes their session, and can be broken down into several distinct functions:

- Identification of all applications relevant to the end user
- Modelling all possible interactions between these applications
- Determining the desired actions to be taken for each of the interactions above

Static interaction management

Static interaction management is where functions are fixed ahead of their execution, whether that execution is before or during the end users session. They rely on unchanging parameters, and the currently standardised mechanisms in the IMS are classified as static interaction management functions. They rely on predictable and predefined service information.

Dynamic interaction management

Dynamic interaction management is where the mechanisms used can be dynamically changed **during** their execution. Examples of this would be where the interaction management functions are based on the **current** context of the end user, the contents of SIP signalling after the initial request or the result from a previous service execution in the same session.

2.2.3 Requirements

The 3GPP identified the following requirements for improvements to the ATA [46]. Any changes must:

- Be flexible enough to handle interactions between new applications
- Minimise the impact of any changes introduced
- Manage service interaction between SIP based IMS ASs
- Perform efficient signalling between the core network and the AS
- Support service integration between different services hosted in the same AS or in different ASs

However, the study was limited in scope and specified that the the following requirements be investigated at a later stage:

- Manage service interactions between different non IMS ASs
- Allow integration between SIP and non-SIP AS
- Support existing IN services (e.g. CAMEL)
- Support integration between multiple service providers
- Allow end users to customise and control their services to provide personalised services

Similarly, they proposed a set of specific service interaction scenarios to study. These are the interactions between:

- The end user and an AS to which the end user subscribes
- Invoked services and services that are still to be invoked

- Services for a particular end user
- Services invoked during a single session
- Services invoked under one service provider domain
- Different ASs

Extensions of these scenarios were left for further study, e.g. service interaction between multiple users, service interaction spanning multiple sessions, service interaction between different service provider domains etc.

2.2.4 SCIM Architecture

Three different types of architectures for service interaction management between the S-CSCF and the AS were proposed by the 3GPP study.

Centralised Service Broker

In this architecture, the S-CSCF and the AS are as standardised. The ASs are unaware of the Service Broker and operate as normal. The S-CSCF views the Service Broker or SCIM as simply another AS and routes messages towards it through the standardised mechanisms.

Distributed Service Broker

This architecture requires a Service Brokering function to be added to each AS. This could be implemented by changing the AS to have this functionality integrated with its core services, or it could be an entirely separate entity that interacts solely between the S-CSCF and one AS. This added functionality handles all interaction between the S-CSCF and the AS. The S-CSCF continues relaying messages between these entities until all service invocation is finished.

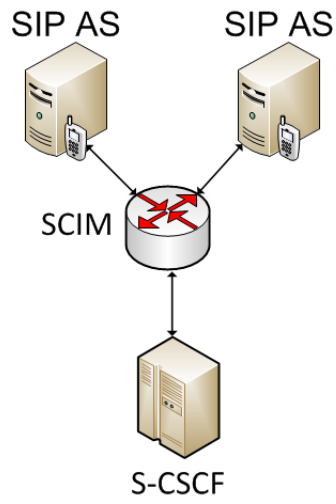


Figure 2.4: Centralised SCIM architecture.

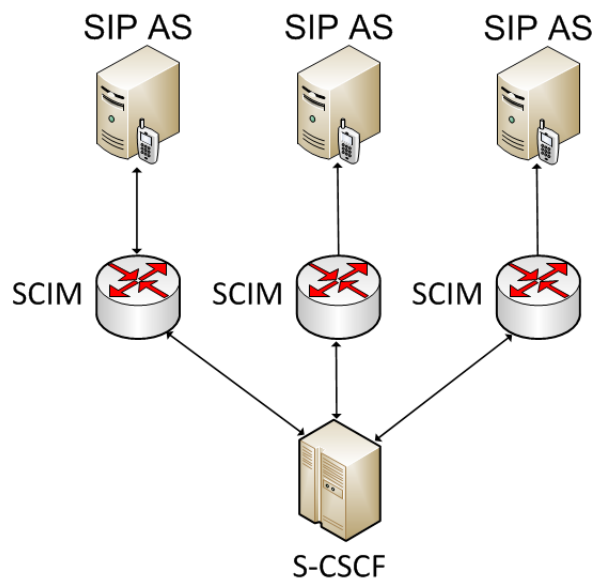


Figure 2.5: Distributed SCIM architecture.

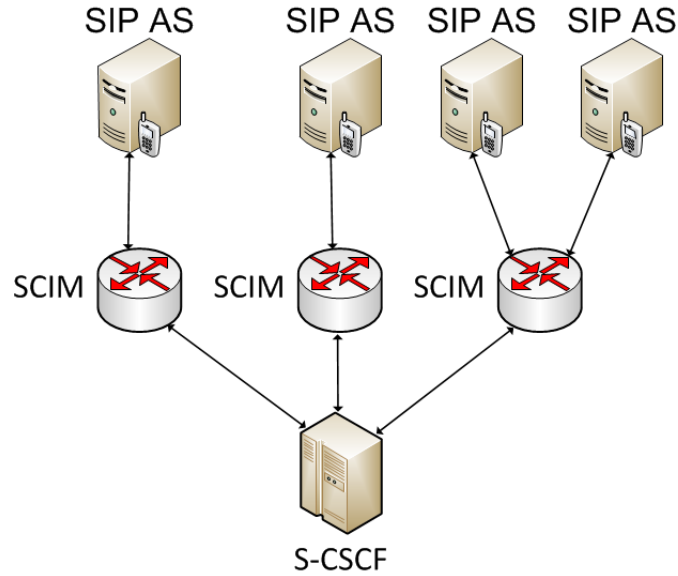


Figure 2.6: Hybrid SCIM architecture.

Hybrid Service Broker

Combining aspects of both architectures above results in the Hybrid architecture. This is an architecture with service interaction management nodes that can manage multiple AS in a one-to-many mapping or a single AS in a one-to-one mapping.

2.2.5 Reference architecture

The 3GPP proposed a reference architecture which allows for the three different service interaction architectures detailed above. It has an ISC interface between the S-CSCF and the AS to transfer SIP signalling messages and an interface to the HSS to transfer service interaction logic. However going forward they standardise on the centralised model to investigate service interaction management issues further. The interface between the SCIM and the HSS has been identified as being optional, and where it is used either the CX interface or SH interface is likely to be used. The decision between

these two interfaces was left for further study. In the general case, multiple ASs will be managed by one central SCIM entity. This entity can either be part of the S-CSCF or a stand alone entity, as depicted in Figure 2.7.

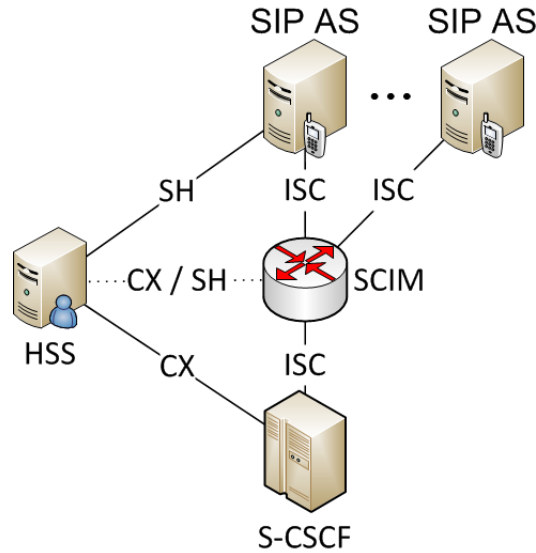


Figure 2.7: 3GPP proposed reference architecture for SCIM.

2.2.6 Functionality

Service History based invocation

The study proposes one high level concept mechanism to allow for dynamic service triggering based on offline service interaction detection. This mechanism utilises a set of rules based on the history of service invocation together with the existing iFC mechanisms. Each service is given a priority as normal, but which service is triggered next depends on the result of the previous service. For example, if service A was executed successfully, trigger service B, else trigger service C. No details are given on offline service interaction detection that could be informed from results of other services which are yet to be triggered.

Equivalent class based conflict detection and resolution

Very little detail is given on this concept. The study proposes classifying services into distinct classes based on their impact to other services or their functionality. However, the study concludes that this functionality is beyond the scope of the investigation and leaves it for future work.

2.2.7 3GPP Conclusions

The proposed reference architecture for service invocation adds the ability to dynamically trigger services based on the result or history of previous services. It does this with minimal changes to the existing architecture by only introducing one new element with compatible interfaces to the pre existing components. The current iFC mechanism is still used. However, modifications to both the S-CSCF and ASs are needed to include the Service Identification and Service History information in the ISC interface. These modifications were not detailed. The 3GPP also started discussing some specific improvements that could be used to target specific problems. These include:

- The separation of the target user information from the served user information, for cases where the AS changes the remote URI in the SIP request
- The concept of comparability classes, where incompatible services are categorised into separate classes depending on whether they would conflict or not, which disallow any service triggering from incompatible classes
- Improving the iFC to allow better handling of error responses generated by the AS
- Adding a new SPT (Service Point Trigger) to include information based

on the end user's terminal capabilities, which are retrieved during registration

- Updating the iFC to allow multiple services from the same AS to be triggered during one SIP transaction

These suggestions are the recommended conclusions of the 3GPP study, and as such do not have any implementation details.

2.2.8 Discussion

As can be seen above the 3GPP has determined that improvements to the ATA are required. These improvements are required because there are several problems with the current standardised architecture, which have been identified by the 3GPP and others [46, 57]. These include the inability to:

- Use contextual information in routing decisions
- Allow the end user to customise his or her service usage according to their particular desires
- Support conflict detection and resolution mechanisms
- Trigger services based on time-outs or after certain responses
- Intelligently handle error conditions

It is not possible to route service requests based on external contextual information about a particular end user. Currently no contextual information is used in the decision making process. This means it is not possible to activate different services depending on external information such as the end users presence or status. The end user cannot change how he interacts with his telecommunication services depending on his current context. An example

of such missing functionality would be to send voice calls to voicemail should he set his status to “*Busy*”.

The next problem is that the service triggering mechanisms are all controlled by the network operator. The end user does not have any control over the iFC and cannot customise his service usage to suit his needs. An example of this would be to allow him to determine what happens when certain people call him i.e. send an email recording the date and time when Alice calls.

In addition there is no ability to trigger services based on time-outs or after certain responses. An example of this would be to allow a call forwarding service to be triggered after a certain time-out which was started on receipt of a SIP 180 RINGING response.

Should any services be triggered that have opposing goals or are in conflict in some way, the end state of the interaction will be unknown. This is undesirable both from the network operators point of view (e.g. charging issues arise) and from the end user’s point of view (e.g. their services don’t work as expected). Mechanisms to control these interactions are needed to determine the proper way to handle such cases in a graceful failure mode.

However, these improvements can not be made in isolation. As there are already pre-existing IMS deployments in the real world, any improvements need to be fully compatible with the current existing architecture, without requiring major changes to existing elements. The improvements need to be designed with a specific set of constraints. These constraints have been highlighted by the 3GPP and others [46, 58]. Any improvements to the service triggering architecture need to bear these constraints and requirements in mind. They should:

- Introduce minimal impact to the existing core architecture and AS
- Allow flexibility to cater for any potential interactions of new services and applications

- Interact with the AS in an efficient manner and avoid redundant signalling
- Permit user service customisation and control

The next chapter discusses several approaches undertaken in an attempt to solve the above problems. As will be seen, there are still unanswered questions around providing the necessary improvements while meeting the highlighted constraints.

Chapter 3

Related Work

While the previous chapter examined the standardised service triggering architecture, multiple research projects have explored new avenues for improving upon it. This chapter describes the different approaches that have been suggested to solve the problems highlighted previously.

Each related work has tackled slightly different areas, although there is some overlap. These different areas can broadly be placed into one of four main functional areas, which can be seen on the following page in Table 3.1. Further details are provided in a section dedicated to each particular feature area. This is followed by a discussion on the different architectures for introducing a SCIM as well as a section on context aware services. The chapter is concluded with an overall comparison highlighting which of the discussed work matches which feature area.

Area	Description
Service Information	Information about each service is needed to solve some of the highlighted problems from the previous chapter. This information can be used to detect and resolve service conflicts as well as optimise service routing.
Service conflict detection and resolution	New mechanisms introduced to detect and resolve any service conflicts. This area revolves around either preventing or resolving undesirable interactions between multiple services.
State Tracking	This area looks at tracking the results of each service interaction and carrying it through to influence further decisions.
Performance improvements	Several different areas of performance improvements have been described in the literature. Either reducing the load on the S-CSCF, the AS or decreasing the amount of signalling required is discussed under this area.

Table 3.1: Main areas of related work.

3.1 Service Interaction Management

The first area to look at is the handling of any interactions between different services. This covers the information needed in the system to identify each service, the potential conflict points as well as actions taken to detect, prevent and resolve service conflicts.

3.1.1 Information required

To properly manage service conflicts, any detection and resolution mechanisms need to have knowledge about the particular services and ASs. The required knowledge has been summarised by Chua et al. to include [58]:

- A list of all available services or the services to to be invoked
- Information about the interface between the SCIM and the service to be invoked
- Relationships between individual services that define how they can interact
- The network operator’s policy or business rules
- End user preferences

Currently there is very little detail in the literature as to how this information can be gathered. In the majority of the related work the information about possible services is supplied by the network operator and manually injected into the system, e.g. as a service interaction database injected into the S-CSCF [59] or an acyclic graph provided by the developer / operator [60]. There is some discussion of using the Service Location Protocol (SLP) [61] to discover services available in the IMS, but the emphasis there is on how the end user’s equipment can discover and utilise existing services, rather than how the network can assimilate new ASs [62, 63].

While different approaches have been suggested for the interface between the S-CSCF, SCIM and AS in terms of functionality and component location, most of the related work has standardised on some form of SIP as per the 3GPP recommendation. There is a form of automatic management of the relationships between services through categorisation based approaches as will be discussed later, but the grouping of services into each group is still done manually.

3.1.2 Service conflict detection

A fair amount of research has been performed with regards to feature interaction management or service conflict detection and resolution. Service conflicts occur when services do not operate correctly when used simultaneously in one session. This is an established problem that has been seen in previous iterations of telecommunication networks e.g. Intelligent Networks (IN) [45]. This has also been researched in various ways for SIP which is the control signalling protocol for the IMS platform [45, 52]. Knowledge and experience from these previous areas of research have driven the research behind several attempts at solving this problem within the framework and architecture of the IMS [43, 64–66].

Approaches that revolve around extending the SIP protocol to prevent service conflicts are also relevant to the IMS platform, as the IMS uses SIP as its signalling protocol. One notable version of this approach has been proposed by Chiang et al. [64]. They propose a SCIM based implementation of the extended service “cocoon” idea proposed by Kolberg et al. in the SIP domain [52]. In Kolberg’s work these cocoons wrap each service with service conflict detection and resolution mechanisms and modify the original SIP request to include various new pieces of information, detailing the service as well as its participants. The additional information added allows the cocoon to detect service conflicts, and resolution is handled by disabling one or more of the services. This information is conveyed in several new headers added to the SIP request that is passed between the SCIM and the ASs.

In the work presented by Chiang et al. the SCIM acts as an intermediary between the S-CSCF and the AS, caching the SIP request message before passing it on to the AS. This cached INVITE request is needed to resolve service conflicts. Once the AS has finished processing a message it would add a specific header, namely a new SERVICE-INDICATION header into the SIP message indicating whether or not the service was actually triggered. This header is stripped from the message by the SCIM before passing it onto the S-

CSCF. When a service conflict is detected a SIP 380 ALTERNATIVE SERVICE message is sent to the service to be disabled if it has already appeared in the signalling path ahead of the service to be activated, and the cached request is sent to the service to be activated. If the disabled service only appears after the activated service the output from that service is discarded by the SCIM.

This approach requires modification to each AS, a new component in the form of a SCIM as well as modifications to the SIP protocol. The signalling path is also extended by the inclusion of services that will not have an effect on the end users session, increasing the session set up delay significantly.

Crespi proposed expanding SIP to include function interaction semantics including the mechanism to use to avoid undesirable service interactions [44]. This was done through the addition of a new SIP header called SERVICE-RULE. This new header is composed of three main parts. The first part, *applicability*, specifies whether or not the rest of the rule applies to the current request or response. The second part, *messagePart*, describes the specific part of the SIP request that is affected by the rule. The final part, *forbiddenValues*, dictates values that must not be set in the part indicated by the “*messagePart*” of the rule. This header is inspected and followed by the AS and not by the S-CSCF. There is no guarantee that a misbehaving AS will follow the rules dictated in the SIP message.

To solve this problem and ensure that all ASs follow the rules correctly Crespi et al. propose adding in new functionality in the form of a SCIM. The SCIM would examine the message before and after passing it to the AS to ensure that the rules have been followed and no unauthorised rules have been added. The SCIM would be able to drop a request that contains forbidden values, but would not be able to repair or continue the session. While this proposal has the advantage of distributing the processing load required to do conflict detection and resolution amongst all ASs it does require modifications to all ASs. It also requires additional work to ensure that each component of the IMS framework (P-CSCF, I-CSCF etc.) as well as the end user terminals

would handle the new SIP header correctly. As such extending the SIP protocol in this way is problematic as there are already large IMS networks deployed [4, 28].

Hua et al. propose modelling the service conflict detection and resolution after an *artificial-immunity-based* online system. They introduce the concept of *Service Behaviour Descriptions* and *Service Interaction Descriptions* which describes the service and its interactions. Constraint rules are used to determine if two services are in conflict, and they introduce a self learning mechanism through the use of a *Constraint Rule Generator* [66]. While this research shows promise for the offline conflict detection, their online conflict detection mechanism shows a very high false positive rate. The work also neglects to mention how it will be integrated into the IMS platform. Similarly, Xu et al. also declare their own language to represent service conflicts and detection, and also use a self learning mechanism to provide some form of online conflict detection [65].

Gouya et al. propose integrating feature interaction management with the S-CSCF [67]. This interaction management element utilises a database of all known service conflicts to prevent services from being activated that will conflict with each other, as well as defining several extensions to SIP that will allow online conflict resolution. These mechanisms rely on the addition of *Service Identifiers* as well as *Service Rules*. While they refer to their conflict detection proposal as having an offline and online component, both mechanisms are activated only during the end users session and as such their proposed architecture is only classed as an online conflict resolution mechanism for the purposes of this thesis. The first part of their solution relies on the SCIM examining the conflict database before each service is activated to determine whether or not it would conflict with already activated services. However, this is limited greatly in that it only allows incompatible ASs to be declared, not individual values or actions taken by the ASs.

To combat this they allow the ASs to add information about future conflicts by adding *Service Rules* through custom SIP header extensions. This allows

the SCIM to determine if the next application server to be invoked is allowed. Details on how the service conflict database is established or maintained were not discussed, and the addition of information increases both the size of the messages as well as the processing time required to parse each message. This adds to the workload of the S-CSCF, which will be identified as the functional element that already bears the largest load further on. The architecture proposed in this thesis reduces the work load on the S-CSCF and provides a way that such a service conflict database can be established and maintained.

Liao et al. also suggest integrating a service conflict database to the S-CSCF, together with the procedure to be followed to resolve such conflicts [59]. Each time a service is invoked the AS's name / address is added by the S-CSCF to the SIP ROUTE header in the SIP request after consulting it's own database to ensure that no conflicts are detected. Their approach is modelled on Gouya's previous work [57] and does not focus further on service conflict resolution. The main focus of their work is reducing the session setup delay, which is discussed further on.

3.1.3 Specific service combinations

One method of avoiding service conflicts is to determine possible service combinations ahead of time that will execute a particular scenario. This also has the added benefit of allowing a more customised experience to be provided to the end user, and can take specific context triggers into account. Goveas et al. utilise a centralized SCIM architecture to allow certain blended services to be offered to the end user [60]. These blended services are represented as an *acyclic service graph* comprised of many different service nodes. Each service node represents a different service or service capability, and there are rules defined for selecting each different flow or path through the graph. These rules take into account context and the service itself, but are statically set ahead of runtime. For example, if the end user's presence was set to *Movie Time* and the end user was not near a movie theatre it would select

a predefined path through the graph such as activating a service to offer information about upcoming movies followed by another service providing a map to the nearest theatre location. This is limited in the sense that routing decisions for the entirety of the service are decided before the session begins, and results from one service cannot influence the triggering of further services.

Thus they have introduced the concept of the SCIM being able to execute different services based on the specific context of the end user, but introduce this as single, specific combinations of services that are designed and provided by a service provider [68]. These are loaded into the SCIM as packaged XML files which contain the necessary service execution path as well as the methods and sources to retrieve contextual information needed for the routing decisions.

To achieve their goals they declare several different blocks of functionality. The first set of functionality discussed is the Data Federator (DF). The DF acts as an aggregator agent which collects information from various different sources. The interface between the DF and the rest of their SCIM is not detailed. The next block of functionality is called the Policy Engine (PE). This takes the context provided by the DF and uses it to select a particular service flow. Next is the Service Broker (SB) functionality, where the specific service flow that was chosen is actually executed. The SB contains interfaces to the various IMS services and will execute the service flow as instructed by the PE. While the discussed architecture does not require any changes to the pre-existing AS a fairly major drawback is the fact that a **“given blended service will be defined by a service developer during service creation or provision time”** [60]. These service flows are set statically and are not flexible. It is not possible to change them during service execution.

Another approach used is the concept of grouping different classes of services into distinct categories, which has been investigated by several different researchers. Nakajima et al. suggest placing services into one of two distinct groups based on whether or not the service terminates the session. The first

group, “*Connection Control Applications*”, is comprised of any service that controls the termination of a session such as 3rd Party Call Control (3PCC) services. The second group, “*B2BUA Co-ordination Applications*”, is comprised of services that do not control the termination of a session such as a call redirection service. Only one service from the first group may be activated during any session while multiple services from the second group may be activated. This does not eliminate all conflicts as it is still possible for services to clash with each other due to conflicting instructions or conflicting changes to the SIP messages but is a starting point for further research [69].

Varatanovic et al. propose a similar approach, but instead of emphasising the type of functionality they place the emphasis on network traffic requirements or characteristics, such as Quality of Service requirements. Each specific group would be served by their own SCIM to prevent non real-time services impacting real-time services [70]. Instead of examining this functionality from a service conflict detection point of view, Zha et al. look at this functionality from a load balancing point of view. They class services into message services (e.g. instant messages) and conversation services (e.g. voice calls). The majority of their work is focussed on how to successfully balance the load between potential future SCIM implementations [71].

3.1.4 State Tracking mechanisms

A drawback of the standardised architecture is the lack of dynamic service triggering based on the results of previous service execution. Wang et al. introduce a User Service Triggering Management (USTM) node as well as extensions to SIP to help the S-CSCF trigger services based on previous results [72]. The results of each service is stored in the resulting SIP signalling, which is examined by the USTM to determine future routing decisions.

3.2 Reducing load and session setup delay

A key factor for the quality of the end user's experience is the session setup delay, i.e. how long it takes to establish a session between the caller and callee. This can either be between a service such as a voice mail service or another end user. Modelling results from Ulvan et al. indicate that a significant portion of delay encountered during session establishment over both UMTS and WiMAX access networks in the IMS is due to processing delays [73].

Xun et al. use an analytic model representing all the factors that contribute to the processing delay in session setup delay to determine that the S-CSCF is the bottleneck in the IMS for control signalling and conclude that further research is needed to improve the standardised ATA of the IMS [74].

This work was followed up by a proposal from Xun et al. to group services into distinct categories or groups to simultaneously trigger multiple sequential services without involving the S-CSCF [75]. In this proposition, the S-CSCF examines the inbound SIP request and builds a list of all matching iFC. It then adds the relevant SIP addresses of each AS to the SIP route header, and forwards the message to the first AS. After the AS performs its functionality, it will forward the SIP message onto the next AS in the route header according to normal SIP routing rules. Assuming that each AS performs a limit set of functionality and does not modify the SIP request this mechanism improves the session setup delay as messages are no longer passed back to the S-CSCF before being routed to the next AS.

However, while this plan does not require any new components to be added to the IMS architecture, nor require modification of each AS, it does not guarantee that the correct service invocation will occur as it is possible for the first AS to modify the request in such a way that it is no longer relevant to other AS. This means that there will now be excessive control signalling as the signalling path has now been expanded to include all the AS that would have served the initial request. The length of the signalling path (i.e. how

many AS are involved in the session setup) has been shown to be a major contributing factor to the session setup delay [59, 74, 76].

Another approach to reducing the session setup delay was suggested by Krishnamoorthy et al. in which they propose adding call control functionality to the S-CSCF to allow it to perform some service logic [77]. This is added through the extension of the iFC to include not only the ability to forward messages, but also to modify the SIP message (including both the headers and body) as well as generating new SIP signalling. While Krishnamoorthy et al. acknowledge that this breaks the rules for the S-CSCF according to RFC 3261 [31] they believe that the additional functionality is needed. While this approach does not require any modifications to the AS, it increases the load on the S-CSCF and does not comply with the pre-existing standards and implementations.

Extending the iFC was also examined by Liao et al. who propose a “*call-state-based service triggering architecture*” (CSTA) to improve the session setup delay [78]. They propose a new set of filter criteria based upon the call or session state, called the “*call-state-based Filter Criteria*” (cFC). These cFC allow different signalling paths to be invoked depending on the current state of the call or service. This evaluation is done on the receipt of SIP requests and SIP responses in contrast to the current iFC which are only evaluated on the initial request. They based their cFC on the iFC and change only the Service Point Triggers (SPT). They further define 4 types of cFC, namely a *session initiation* type, a *provisional response* type, a *final response* type and lastly one to be used on *session termination* (e.g. SIP BYE message). When a SIP request or response is received by the S-CSCF only the relevant cFCs are queried. This reduces the signalling overhead and reduces the session setup delay, but increases the processing needed at the S-CSCF. Their proposed improvements required a heavily modified S-CSCF but otherwise do not require any further changes to the existing architecture. The work undertaken by both Xun et al. and Liao et al. was expanded and merged to create a “*linear chained approach for service invocation*” for the IMS [59]. This is a more detailed proposal which provides details and

algorithms for implementing the ideas mentioned in the previous works, as well as models showing a reduced session setup delay. The mechanisms for handling service conflict detection and resolution introduced in this merged work has been discussed in the previous section.

Chiang et al. also propose removing unnecessary ASs from the signalling path [79]. They propose a *Distributed Service Invocation Function* (DSIF) which is overlaid over the traditional IMS architecture. It is composed of three different nodes which are added to an existing IMS deployment.

The first node is called the *Core-network Service Invocation Function* (C-SIF). This part interacts with the existing S-CSCF component and acts as a gatekeeper, only forwarding messages to the AS when the service represented by the AS will be executed. This is possible as some services such as third party call control [80] might only be interested in the “final response to identify success or failure of the [session] establishment” and will ignore provisional responses [79]. The C-SIF can also immediately return a response for notification messages to the S-CSCF while it is busy forwarding the notification message to the desired AS. In addition to this, the C-SIF inspects all messages, allowing it to do service invocation not only on the initial SIP request but during any stage of the session. In order to do this, they introduce two new types of filter criteria, namely the *trigger Filter Criteria* (tFC) and *execution Filter Criteria* (eFC).

These FC are stored in the second new node, which is node called the DB-DSIF. The last node used in the new architecture is the *Application-side Service Invocation Function* (A-SIF). This node is used to handle all interactions with the AS. They introduce a mapping of their own control messages to standard SIP messages, allowing the C-SIF to communicate with the A-SIF over the standard SIP interfaces. They tunnel the communication between the S-CSCF and the AS inside different SIP messages, with their own control messages forming the content of the standard SIP messages. Their system has the advantage of reducing the overall session setup delay for certain cases without requiring modification to any existing IMS components. How-

ever, each AS is handled by the A-SIF acting as a back-to-back user agent (B2BUA) [31]. This limits the flexibility of the system and hampers cross AS service combinations, and does not allow for any form of service conflict detection and resolution.

A radically different approach to reducing the session setup delay was proposed by Al-doski et al. [81]. Here they propose an intelligent service selection algorithm to provide an optimal set of services to meet certain delay guarantees, with the aim of maximising revenue for the operator by providing the largest set of services that can be successfully handled given certain resource constraints. Rather than reduce the session setup delay per session they focused on preventing sessions with session delays that may be larger than set targets being allowed unless the network has the necessary resources (processing power, bandwidth etc.) to guarantee a satisfactory performance.

Qi et al. design an architecture that removes the S-CSCF from the signalling path between each AS, and instead allows the AS to communicate directly between them [76] in a similar vein to the work discussed above by Xu et al.

However, in their design, modifications are needed to each application server as well as the addition of a central control point. This also requires that each Application Server (which could be provided by a non trusted entity) operates correctly and passes on the message without tampering. In addition this architecture will only work for services that do not modify the message or destination. The routing decision as to where the message will next be sent is made in parallel with the execution of the service logic, i.e. before the service logic has finished executing. This is not a viable solution for many call control services e.g. call forwarding services.

Liao et al. improve on this architecture by evaluating all the initial filter criteria (iFC) simultaneously and then modifying the SIP message to include a custom *Route* header indicating the route the SIP message should take [59]. This reduces the complexity of the customisations required for each Application Server as it should already have the necessary functionality to

follow directions specified in the SIP Route header and should not require further modification. However, it is still not possible for the routing of new service requests to change after one service has been activated. Their proposed architecture reduces the session setup delay. However, it also reduces the flexibility of the architecture as SIP messages can no longer be forwarded to other ASs depending on the results of their own service execution.

3.3 SCIM Architecture

The reference architecture proposed in the previous chapter by the 3GPP highlights the addition of a SCIM between the S-CSCF and the AS. However, this is from a logical view. The actual functionality can reside in either the AS itself, the S-CSCF or in a stand alone element [58].

3.3.1 Internal as part of AS

Placing the SCIM functionality inside the AS would require changes to be made to each AS, and is closest to the “*Distributed Architecture*” shown in the previous chapter. Each AS would need to know about the services and capabilities of the other ASs in order to perform any form of service conflict detection and resolution. This would require redeployment of the individual AS as well as some supporting infrastructure to make this knowledge available to the ASs. The implementation of any service co-ordination functionality would be dependant on the technology used in each AS. For example, it is likely that it would be controlled by SIP as the traditional AS understands SIP. A different implementation would be required for each type of AS that was developed. This also limits the flexibility of each individual AS, as should the service change, redeployment of all ASs could be required to accommodate new service rules or logic.

3.3.2 Internal as part of S-CSCF

A different approach would be to integrate the new functionality into the S-CSCF. This approach as well as the approach described in the following section (stand alone component) would fall under the “*Centralised Architecture*” as seen previously. This proposal would require modifications to the S-CSCF, but should not require changes to the individual AS. However, introducing extra functionality into the S-CSCF would increase the load of the S-CSCF. This should be avoided as the S-CSCF has already been identified as the bottle-neck in the IMS signalling plane [76, 78, 82]. Integrating additional functionality into the S-CSCF would not have as high a technology dependency as integrating it into the AS as only one implementation would be required for a particular network deployment. Changing this functionality or redeploying it could prove challenging as the S-CSCF is a central element to the IMS network. It is not possible for the IMS network to function without a S-CSCF.

3.3.3 External as stand alone component

The last approach discussed is the introduction of a stand alone function to perform service interaction management. This approach has the benefits of not introducing more load to the S-CSCF as well as allowing it to be deployed into pre-existing IMS networks without requiring modification to either the AS or the IMS components. It allows for a clean break of functionality between the S-CSCF, the SCIM and the AS. This also allows easier service redeployment, as only one element needs to be updated with new service interaction rules. As there is already a standardised interface between the S-CSCF and the AS the same interface can be used for the SCIM. An additional factor in favour of the stand alone SCIM is the fact that it represents a central point for the application of control policies as well as a point where user-centric information can be mined for potential trends etc [60].

3.4 Context and context aware applications

There has been a large drive to increase the intelligence in our applications and services to allow them to make the correct decision or offer the appropriate functionality depending on the specific environment or context that the end user is in [83]. At a bare minimum, the context of the end user can be mapped to the concept of the end user's *presence* or *status*. Presence is often used to refer to the “*willingness*” or “*availability*” of an end user to communicate with another user.

Other contextual information would include the location of the end user as well as the capabilities of the device the end user is using. Various non IMS based context aware services have been proposed in the literature, both in terms of the end user's status or presence [84, 85] as well as in terms of the context of the end user's device (reading from its sensors etc.) [86, 87].

3.4.1 SIMPLE Presence

The Session Initiation Protocol for Instant Messaging and Presence Leveraging Extensions (SIMPLE) is a suite of protocols and extensions based upon SIP. This heavily leverages “*A Presence Event Package for the Session Initiation Protocol (SIP)*” (RFC-3856) [88] and “*SIP-Specific Event Notification*” (RFC-6665) [89]. There are multiple different RFCs for the various subsets of functionality exposed under SIMPLE based presence systems. The presence architecture used in the IMS is modelled on the SIMPLE presence system, with a few modifications which are explained in the next section.

There are two main roles defined in SIMPLE presence. The first role is the role of the *presence entity* or *presentity*. This is the role performed by the person who is sharing information about their own context. The presentity supplies a certain set of information (status, device capabilities etc.) through the use of *Presence User Agents* (PUAs). One presentity may have one or

more PUAs, each contributing different information about the context of the end user or the capabilities of the devices they run on. These PUAs send the presence information to a *Presence Agent* (PA) which is responsible for amalgamating all the different pieces of status into an overall view of the end user's presence.

The second role is known as a *watcher* role. This is the person or device that is interested in monitoring the changes in status of the presentity. It is possible to not only request the current status of a particular end user, but also to subscribe to the end user's status. This is done through the use of a SIP SUBSCRIBE request, and the watcher will be notified of any changes in the presentity's status through the use of a SIP NOTIFY request. The watchers may also filter the information they are interested in by modifying the SUBSCRIBE request, which will limit the notifications they receive to only the specified context changes (providing they are allowed access to the requested information) [90]. These SUBSCRIBE / NOTIFY transactions include a special value in the EVENT header which identifies which particular type of event the transaction refers to. For presence based systems this value is set to **presence** as standardised by RFC 3856 [88].

The actual presence information is kept separate to the SIP signalling and is normally represented by an XML body attached to the NOTIFY request. This XML body is formatted according to the *Presence Information Data Format* (PIDF) [91]. The base PIDF defines a basic format for conveying presence information including a text based entry, indication of availability (*open* or *closed*) and a specific URI.

The *Rich Presence Extensions to the Presence Information Data Format* (RPID) extends this base format by adding several new elements to communicate additional information such as what the end user is currently doing, where the end user is currently located, the end user's current mood etc [92]. There is also an emphasis placed upon non human agents being able to create or parse these elements in comparison to the *note* element of PIDF which is intended for human operators.

Another commonly used extension is the *Contact Information in Presence Information Data Format* (CIPID) [93]. This extension allows the presence system to communicate additional information about a presentity, such as a link to “*business card*” information, homepage, map of location etc.

Similarly the *Timed Presence Extensions* to the PIDF conveys additional information about when and for how long various presence attributes are valid [94]. Of particular relevance to this work is the “*Session Initiation Protocol User Agent Capability Extension to Presence Information Data Format*”. This format is used to express the capabilities of the end user’s User Agent (UA) in a fairly detailed manner. It allows for an indication of what types of services it can handle (e.g. audio, video, text, data etc.), whether or not it represents a human or machine operator, whether it can operate in full-duplex or only half-duplex mode, the SIP request types it supports etc.

These extensions, together with a new proposed extension format for communicating service information in the IMS, are used in the work presented in this thesis.

When a presentity wishes to update their status their PUA sends a SIP PUBLISH request to the PA or a *Presence Server* which is a PA with some added functionality [95]. This can be initiated by the end user (e.g. changing their status to “*Busy*”) or by some automatic measure (e.g. a device detecting the user is currently not present and changing their status to “*Idle*” or “*Away*”).

The PUBLISH request contains a presence document which is merged together with other presence documents received at the PA according to certain composition rules. The presentity can also upload a rule or policy document which dictates which “watcher” is allowed access to which information.

For example, certain watchers may be able to see specify information such as current location, whereas other watchers are only allowed to see availability information [96]. An example protocol that can be used to control this policy document is XML Configuration Access Protocol (XCAP) [97].

XCAP allows the end user to upload, download and modify an XML document utilising HTTP. The end user could also use XCAP to manipulate a particular presence document which could form one of the inputs to the PA's amalgamation process [98]. This allows presence information to be conveyed from devices that do not support SIP. Similar to the SIP NOTIFY requests, PUBLISH requests can be used to convey a wide range of different event states and the addition of the “**presence**” value in the “*Event*” header distinguishes the presence use case from other PUBLISH messages.

This section has given a brief overview of the Presence architecture as standardised for use by SIP based applications over the internet. There are numerous optimisation strategies used in real world implementations, such as Resource Lists which allow an end user to subscribe to multiple different PAs through the use of an intermediary service [99] or the use of partial notifications that only communicate changes in presence information and not the full XML document [100]. These have been excluded for the sake of brevity. The next section highlights how this presence system has been incorporated into the IMS architecture.

3.4.2 Presence Architecture in the IMS

As SIP is the protocol used for control signalling in the IMS, it makes sense to examine how the SIP presence infrastructure could be adapted to the IMS. The 3GPP have taken advantage of the RFCs mentioned above to standardise the presence service in 3GPP TS 241.141 [101].

However, the presence service standardisation effort has largely been moved under the Open Mobile Alliance (OMA) as it is seen as one of the core *enablers* or *building blocks* for innovative service creation. The main presence specification under the OMA is the OMA Presence SIMPLE V2.0 Enabler Release [30]. These standards overlay the SIMPLE presence infrastructure over the IMS architecture.

As such the end user's device acts as both a watcher (so they can receive status information about other IMS users) as well as a PUA (so they can publish their own status information). A Presence Server (PS) is added to the standard IMS architecture to act as a Presence Agent (PA). Several additional components can be added such as a Content Server and a Presence XML Document Management Server (XDMS).

All these components are standard IMS SIP based ASs and communicate with each other and the end user using SIP for real time presence data and XCAP for policy configuration. The PS accepts, stores and distributes presence information. It does this by receiving and processing incoming subscription and publish requests from the end user as well as generating notifications. The PS also integrates presence information from the Presence XDMS. It can store or retrieve objects from the Content server should additional MIME based resources be required.

All the SIP signalling is carried out over the standard IMS core. An important distinction to note is that AS can act as watchers themselves. This functionality is normally integrated into a different service that has some presence based state.

When the end user's terminal launches its presence application (normally on registration), it sends a SIP PUBLISH request to the P-CSCF containing a PIDF XML document with the Event header set to "presence". This message is then forwarded to the S-CSCF where it is examined per the normal IMS application triggering architecture. Should a PS be located within the IMS network the S-CSCF will have a iFC configured to match these PUBLISH requests. For example, the iFC may state that all incoming SIP requests with the Event header of "presence" be forwarded to the specific AS that is acting as the PS. The PS will then authorise this request and reply with a SIP 200 OK response message. Similar signalling paths will be followed for the SUBSCRIBE and NOTIFY requests which can be seen in in Figure 3.1.

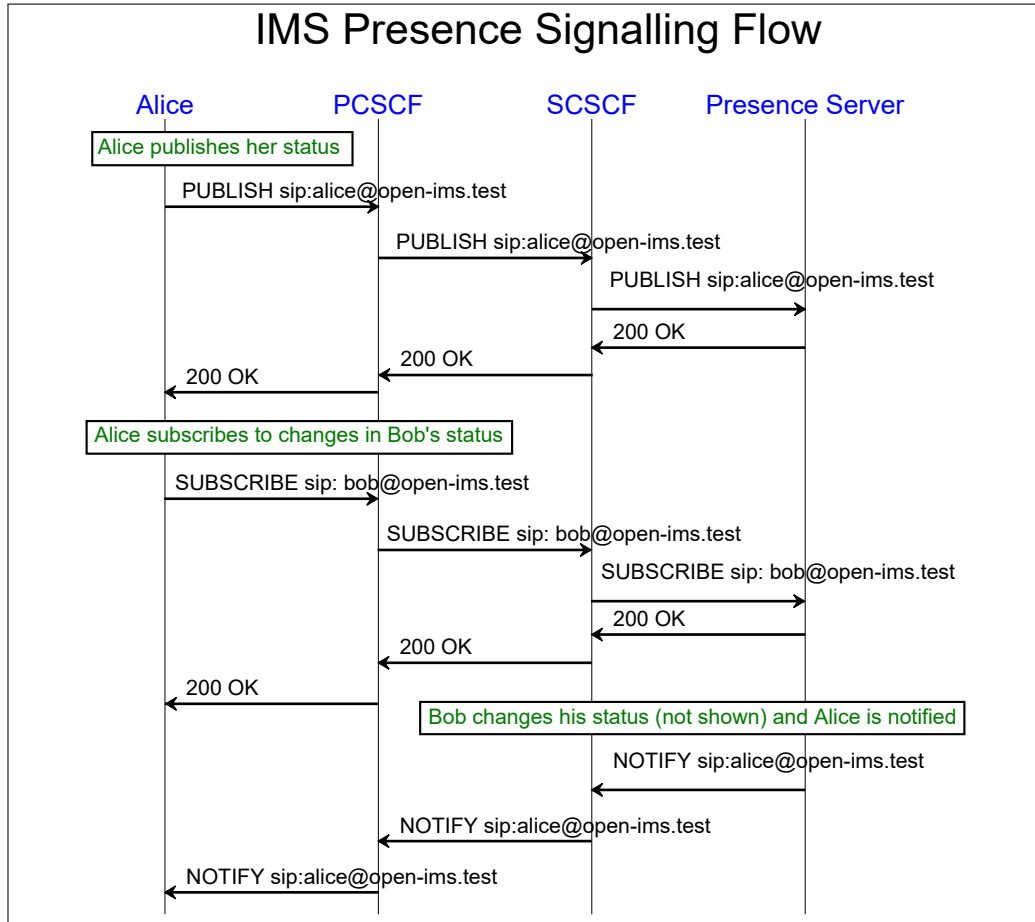


Figure 3.1: IMS Presence Signal Flow.

3.4.3 Example context aware services

Zhou et al. have proposed a “*Context-Awareness Service Adaptation Mechanism*” for the IMS which allows for the adaptation of individual services depending on the end user’s contextual information [102]. This is done through the introduction of an Adaptation Manager which evaluates the end user’s context (including device capabilities, location and network characteristics) and provides a service that best matches the end user’s current context. For example a lower quality video stream would be presented to the end user if their device capabilities reflected a lower screen resolution or poor network

signal. The main focus of their work is to “find an intelligent solution to the service adaptation problem based on collaborative users” [102]. Their work revolves around the intelligence (algorithms) needed to automatically infer the best service from a set of contextual information, the user’s previous history and similar usage patterns.

Another use case for a context aware SCIM is suggested by Sunku et al. [103]. Here they propose using the SCIM as an intelligent agent that is context aware. It can examine inbound messages for certain services and trigger other actions based upon the message content, such as fetching information about the service from the AS as well as its users, and inform the end user of these in a separate session. The example use case given was the SCIM informing a particular end user of which users were currently playing an online game when that end user requested a new game. Most of their work revolved around the charging for such a service and in depth details of the SCIM functionality were not supplied. The SCIM functionality was based upon work done by Goveas et al. which has been discussed in the previous section [60].

3.5 Discussion

This chapter has covered several different areas. The first section covered the related work proposing improvements to service orchestration in the IMS, including service conflict detection and aspects related to the overall performance of the system. This was followed by a general discussion around possible architectures for introducing a SCIM, together with a look at the relevant context related work with a view to introduce context aware service routing later on.

The notable improvements to service orchestration in the IMS that have been suggested in the related work have been summarised in Table 3.2. As can be seen from the table, while there is significant work in this area there is no

Related Work	Service Interaction	Performance	State Tracking	Service Information
[44] Crespi et al.	✓	✗	✓	✓
[58] Chua et al.	✗	✗	✗	✓
[59] Liao et al.	✓	✓	✗	✗
[60] Goveas et al.	✓	✗	✗	✓
[64] Chiang et al.	✓	✗	✓	✗
[65] Xu et al.	✗	✓	✗	✓
[66] Hua et al.	✗	✗	✗	✓
[67] Gouya et al.	✓	✗	✗	✓
[69] Nakajima et al.	✓	✗	✗	✓
[71] Zha et al.	✗	✓	✗	✓
[72] Wang et al.	✗	✗	✓	✗
[73] Ulvan et al.	✗	✓	✗	✗
[75] Xun et al.	✗	✓	✗	✓

Table 3.2: Comparison of service orchestration related work.

single unifying architecture that tackles all highlighted areas.

These different approaches, together with the requirements and constraints discussed in the previous chapter, informed the design of our proposed architecture which is discussed in the next chapter.

Chapter 4

Proposed Design

The previous chapters have discussed the state of art for service orchestration architecture in the IMS and found deficiencies with the current approaches. This chapter proposes a new architecture and framework that builds upon this work. The architecture does this in an novel way while remaining within the requirements discussed. The first section gives a high level overview of the design considerations and requirements undertaken in this work. This is followed by the proposed architecture and framework, focusing on the overall functionality. The chapter is concluded by a short discussion highlighting the problems that are solved by the new framework.

4.1 Design Considerations

Several specific deficiencies were found through the literature study. However, this work does not take place in a vacuum. Pre-existing environments need to be considered, and care must be taken to ensure that the proposed architecture is feasible for real world deployments. This section examines these considerations in closer detail.

4.1.1 Design requirements

Any architecture or framework presented to solve the challenges faced in this area needs to meet several requirements as discussed previously in Chapter 2. The key points have been summarised here to give structure to the rest of this chapter.

- There are multiple live IMS deployments around the world, and as such any changes should introduce minimal changes to the existing core architecture and ASs.
- The changes must not limit the flexibility of the existing system. Instead it is preferable to rather increase the flexibility of the ATA to allow more nuanced service orchestration and to allow new scenarios.
- The new service orchestration framework should allow end user control. Allowing the end user to tweak and customise their service usage should improve their overall experience.
- The framework should take into account the possibility of having conflicting services activated at the same time. It should allow for some form of service conflict detection and resolution.
- Any changes to the architecture should remain cognisant of the need for having multiple different services activated at the same time, i.e. it should not hinder a combined service that has been constructed from individual services.
- Lastly, the improvements should strive to remain as efficient as possible and avoid redundant signalling between the AS and core network.

4.1.2 Core component integration

The work discussed here proposes a novel framework that is intended to be added to pre-existing IMS deployments. As such it should integrate with the

common IMS elements as standardised by the 3GPP. The most important elements here are the P-CSCF, the I-CSCF, the S-CSCF as well as the HSS. These elements have already been covered as part of the standardised IMS architecture in Chapter 2. The work presented integrates fully, requiring no modification to the original network components.

4.1.3 Service components

In addition to integrating with the non modified core components of the standardised IMS, the proposed framework needs to integrate with standard SIP ASs. Any design proposed would ideally not alter the SIP ASs at all.

However, small modifications may be needed to utilise all aspects of the proposed framework. Any modifications must be limited as much as possible and should not result in altering the AS beyond cosmetic changes. Specifically, new interfaces and control protocols must be avoided as this increase the complexity of the AS and increases the required time to develop and test any changes, while increasing the burden required to maintain the service landscape should non SIP based protocols be utilised.

This thesis only proposes one optional change to the ASs. This is the addition of a standard SIP message that contains information about the service. This requires no modifications to the service logic, no new interfaces or protocols, and as SIP is already used for the service's control signalling it should already have the necessary protocol stack to send this message.

4.1.4 Client components

Lastly it is necessary to look at the end user equipment to ensure that the design principles are sound, as the work contained in this thesis involves allowing the end user to control their telecommunication services and customise their user experience. In order to evaluate the proposed design, the

client software needs to be able to register on the IMS network as well as change their context in a meaningful way. This allows us to determine that their service triggering can be influenced by this context change. The client software will also need to be able to utilise several different services to examine how they interact with each other. Services that are expected to be available by default include instant messaging, voice and video calls, and some form of contact management service.

4.2 New functionality

The proposed framework has been built upon three main concepts.

The first concept can be summarised as introducing a form of information gathering, including information about the services available in the network, the status of the core network as well as the context of the end user.

The second concept revolves around the actual changes necessary to dynamically route messages (and thus trigger specific services) according to certain rules.

The last concept is concerned with the new mechanisms introduced to allow the end user to control and customise their service triggering. These concepts are expanded upon in the next sections, followed by the presentation of the actual architecture used to achieve these concepts.

4.2.1 Information Gathering

Certain information is required to enable service conflict detection and resolution mechanisms, as well as dynamic service orchestration. Some examples of such information include the:

- List of available services

- Capabilities of the services (multimedia handling, SIP processing details etc.)
- Relationship between different services

Traditionally this information has been entered manually into any system (standardised or proposed in the literature) by the network operator. However, this leads to a large management overhead as the number of services is predicted to expand rapidly. Manually entering the details and ensuring they are kept up to date becomes infeasible as the number of third parties and services increase.

Thus it is proposed that an automatic way of gathering this information is required. In the proposed framework this information is supplied automatically by each AS. When each AS or service first becomes available, it registers with the framework and communicates its information through a mechanism based on the standard IMS presence functionality.

This mechanism can also be used to indicate current service information such as resource utilisation, either communicated on certain events or after a set period of time. The information is communicated to a central location where it can be made available to the rest of the framework.

A more general version of the service information is made available to the end user. This allows the end user to view information about each service and make decisions on how they want their services to behave. The resource monitoring information is exposed to the network operator to allow them to observe the behaviour of each application service as well as the health of the overall platform.

4.2.2 Dynamic Service Routing

Increasing the flexibility of the IMS ATA to meet the increasing requirements and needs of the IMS platform requires a more dynamic approach to request

routing. The majority of the previous work in this area results in static service execution flows that are either determined ahead of time during service deployment and provisioning, or during the first examination of the initial SIP request by the S-CSCF. While these are improvements on the standardised architecture, more flexibility is required to achieve the goals of the IMS and improve the end user's experience. Previous work in this area has contained the following limitations:

- Extensions to SIP or modifications of the original SIP messages [72, 77]
- Deployment of individual blended services made up of multiple individual service capabilities which are executed as one service through preconfigured service graphs defined by the service developer [60]
- Modification of the S-CSCF to extend the initial Filter Criteria to include more parameters and actions to be undertaken, as well as examination of the Filter Criteria by the S-CSCF on every message [78]

The new framework is able to route specific messages to different destinations depending on both internal and external sets of information, allowing a much more flexible service orchestration environment while also providing the necessary pieces to perform service conflict detection and resolution.

It does not require any modifications to the S-CSCF, does not require extensions to SIP, does not modify the individual SIP messages through the use of additional parameters, and allows actions to be taken not only on the initial SIP request but also on SIP responses as well as timed events. It utilises normal SIP messages out-of-band with the individual sessions to achieve these goals.

The resulting service flow is more flexible than previous work seen in the literature, as the decision on where to route the SIP request next is taken at multiple steps during the session. This is in contrast to the routing decisions being completed only once at the beginning of the session as highlighted in

the literature review. This allows the result of each service's execution to influence the next service triggered. The framework also allows the routing decisions based on the context of the end user, as well as the end user's specific instructions or desired service execution.

4.2.3 User customisation

The last core set of functionality introduced in this framework is functionality to allow the end user to customise and control their service usage. An interface between the end user and the framework is added, which allows them to build customised service execution scenarios. These customised scenarios are transmitted to the routing engine to be executed when specific conditions are met. It is expected that the network operator would create default policies that would be applied to all users, but allow individual users access to tweak and customise their service usage to their own particular needs. This allows the end user to specify how their services interact and what services they want to activate under which conditions. For example, they might wish to redirect all calls to their Voice Mail service when they have set their presence to "*Busy*", or receive their SIP messages as emails by activating a message gateway service when their presence indicates they are using some form of voice only device.

4.2.4 Efficient control signalling

The proposed framework does not result in additional redundant control signalling. It is not possible to build up a signalling path that includes redundant ASs as could be the case with the work presented by Crespi. Here the ASs were still involved in the signalling path even if they were not allowed to perform the requested action [44]. The proposal made by Chiang et al. results in inactive services being involved in the signalling path, as well as extraneous messages that need to be discarded [64]. The standardised

ATA gives the IMS flexibility in handling its SIP based services, but each AS introduced into the signalling path introduces a delay, and as such additional services in the signalling path should be kept to a bare minimum. As the number of services involved in the session increases, this delay increases, which has significant performance implications [74, 78, 82].

Additionally, in a fair amount of the related work the routing of each service request is still handled by the S-CSCF. The S-CSCF has been shown to be a bottleneck when analysing the performance of the IMS SIP control signalling. The proposed architecture offloads part of the routing processing load to the SCIM allowing new functionality without increasing the existing burden on the S-CSCF. It thus allows a reduction in the S-CSCF workload while maintaining backwards comparability with existing IMS deployments.

The related work on reducing the session setup delay and load on the S-CSCF has revolved around optimising the signalling between each AS and the S-CSCF - in some cases removing it entirely [59, 75]. However, these approaches limit the functionality of the service triggering architecture. They rely on taking the S-CSCF out of the signal path by fixing the message routing during the initial examination of the request. The proposed architecture does not go to this extreme as the additional functionality and flexibility is valued over reducing the session setup delay in such a way.

4.2.5 Service description language and service control

Several of the related work discussed in the previous chapters have their own terms used to describe services as well as their behaviour for purposes of service conflict detection and resolution. Examples include new SIP headers with names such as *service-rule* or *forbidden-values* or terms describing behaviour such as *Service Behaviour Descriptions* and *Service Interaction Descriptions*. This work is no different, and has proposed its own service description language, detection and resolution mechanisms together with the architecture enabling these mechanisms.

While the work proposed in this thesis describes similar service control logic based on the contents of the SIP requests as found in the related work by Crespi et al. [44], it does not rely on expanding the SIP protocol. It does not require the AS to take part in the service conflict detection and resolution mechanisms. Instead this logic is controlled by the SCIM. Information is transferred out-of-band of the session, utilising existing standard SIP request / responses. The information is contained in the body of the request as an XML document, and the CONTENT-TYPE header is set to APPLICATION/SERV_DESC+XML. This information contains details such as:

- SIP Headers read by the service
- SIP Headers modified by the service
- Service Meta-Data (Service Name, Version Number etc.)
- Service Capabilities (Duplex multimedia handling, audio only etc.)
- Supported SIP request methods (SUBSCRIBE, OPTIONS, INVITE etc.)
- Resource Utilisation (Free memory, number of users served etc.)

An example of such a document can be seen in Figure 4.1.

The XML document and mechanism used to transfer the information are based upon the Presence Information Data Format (PIDF) [91] and the presence mechanisms found in the IMS [30]. PIDF is a standard that has been extended multiple times to add new functionality through custom extensions, such as the “Rich Presence Extensions to the Presence Information Data Format” (RPID) [92] and “Timed Presence Extensions to the Presence Information Data Format to Indicate Status Information for Past and Future Time Intervals” [94]. Most notable amongst these extensions is the “SIP User Agent Capability Extension to Presence Information Data Format” [104] which details the XML format used to describe the capabilities of the end user’s device, including whether or not it supports audio etc. The idea used

here applies the same concepts, but applies them to IMS services instead of SIP user agents.

```
<Service>
  <Service_Information>
    <Name>Voice Mail</Name>
    <Type>Session Establishment</Type>
    <Version>0.4</Version>
    <Provider>Telco Voice Services</Provider>
    <Description>This service will let the caller record
    a message which can be retrieved later by the callee
    </Description>
  </Service_Information>
  <Service_Config>
    <Server_IP>192.168.0.150</Server_IP>
    <Server_Port>7890</Server_Port>
    <Server_URI>sip:voicemail@open-ims.test</Server_URI>
    <GUID>71f7523d-b461-405b-ab14-695014e03fd4</GUID>
  </Service_Config>
  <SIP_Headers>
    <READ>FROM</READ>
    <WRITE></WRITE>
  </SIP_Headers>
  <SIP_Responses>
    <SIP_200_OK>Successfully accepted call,
    will begin recording callers voice
    </SIP_200_OK>
  </SIP_Responses>
  <Capabilities>
    <audio>true</audio>
    <video>false</video>
    <duplex>recv-only</duplex>
    <methods>INVITE</methods>
  </Capabilities>
  <Metrics>
    <TotalCPU></TotalCPU>
    <CPU></CPU>
    <TotalMemory></TotalMemory>
    <Memory></Memory>
  </Metrics>
</Service>
```

Figure 4.1: Service Configuration XML.

4.3 New Architecture

There have been several different architectural designs proposed in the literature, both from individual research teams as well as from the actual standards body responsible for the IMS. This work presents several new additions. These are the Context Server (CS), the Service Repository Server (SRS), the User Service Policy Server (USPS) and a new version of the Service Capability Interaction Manager (SCIM).

The CS is used to gather contextual information about the end user which is then transmitted to the SCIM to influence its routing decisions. The SRS receives information about each particular service and keeps an up to date record of what services are available, their various parameters and their current status. This information is also transferred to the USPS which allows it to offer the end user the ability to create custom service usage scenarios. The SCIM is responsible for merging all these different sets of information together and dynamically routing an end user's request to achieve the desired outcome. These new elements can be seen in Figure 4.2.

4.3.1 Service Capability Interaction Manager

As detailed in Figure 4.2, a centralised architecture model has been chosen, with the SCIM separate from the S-CSCF. The enhanced routing decision making is offloaded from the S-CSCF to the SCIM. This allows the workload of message routing to be split between the S-CSCF and the SCIM. It is possible to forward all requests or only particular requests to the SCIM depending on the desired deployment. For example, it is envisaged that only users that have either subscribed to particular services or have customised their service usage will have their requests offloaded to the SCIM through the current standardised iFC, allowing the S-CSCF to continue routing standard requests.

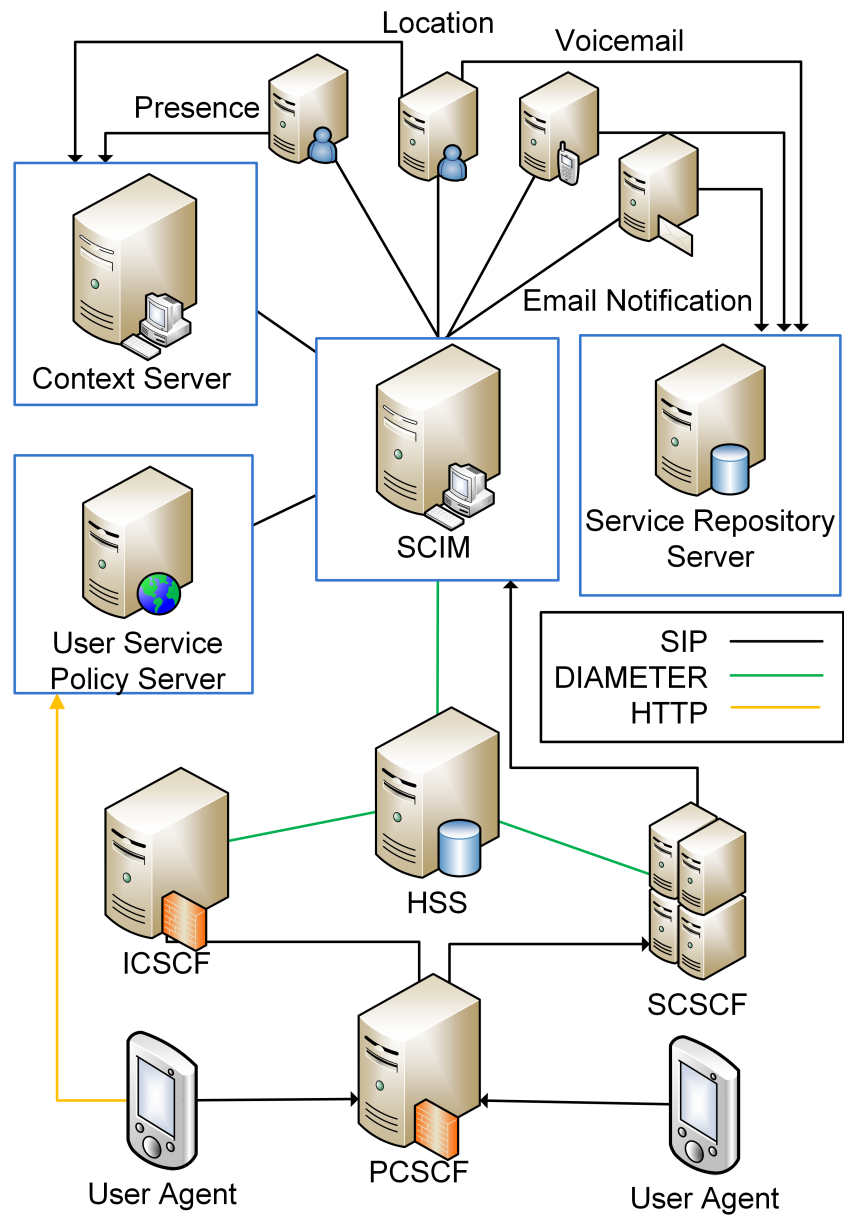


Figure 4.2: Architecture for proposed framework.

This results in the workload being reduced at the S-CSCF which has been identified as the current bottleneck of the standard IMS architecture in related work. The SCIM acts as the intelligent routing agent in the proposed system. It controls the flow of SIP signalling for any enriched services. It makes these decisions based on the contextual information that it has received from the CS together with the desired scenarios that have been created and transferred from the USPS. This information is transferred through the use of SIP MESSAGE requests to the SCIM. The motivation for the separation of the SCIM and the S-CSCF has been detailed in the sections below.

Performance

Firstly, separating the SCIM from the S-CSCF was viewed as necessary from a performance and load balancing point of view. The S-CSCF has been identified as the core element with the highest processing load and plays a significant role in the session setup delay [71, 73, 74, 105]. Adding more functionality to the S-CSCF (i.e. the service orchestration mechanisms in the SCIM) would increase the amount of processing needed per message, increasing the workload of the S-CSCF significantly. This would have a direct impact on the session establishment delay, which is important to the quality of the end user's experience.

Reliability and fault tolerance

Secondly, there is the matter of reliability. Allowing the SCIM to be a separate entity results in a more robust design as it allows the graceful handling of a failure in the SCIM component. Should the SCIM component fail, messages can still be processed by the S-CSCF and only the advanced functions offered by the SCIM will not be available. If they were integrated as one component from an architectural point of view a failure in the SCIM would affect the S-CSCF and vice versa. This separation increases the complexity

of the architecture by adding a new element with interfaces between the existing S-CSCF and existing ASs, but increases the reliability of the overall design.

Backwards compatible and least invasive

The third reason to consider this separation is the requirement for the new framework to be backwards compatible and require the least amount of changes to existing IMS deployments. Integrating the SCIM with the S-CSCF would require modifying or even replacing the S-CSCF. This poses challenges for the network operator to make such a large change to pre-existing IMS deployments.

However, should the SCIM be a new entity it is easier to deploy into an existing network. The SCIM can be deployed in an existing network without any modifications to pre-existing components, and the normal IMS iFC can be used to forward specific requests to the SCIM. This allows a gradual transition to the new enriched functionality with the ability to keep existing users and services unaffected.

4.3.2 Context Server (CS)

The Context Server (CS) is the single point of contact for the rest of the framework to receive contextual information about the end user. The CS gathers information from a variety of sources (e.g. end users' presence, location etc) and amalgamates it for transfer to the SCIM. The timing of this transmission can be controlled through various means, including a time based approach or a batch size approach.

Architecture

In the proposed architecture the CS is a stand alone entity. Previously this functionality was mentioned by Goveas et al. as functionality that forms part of the SCIM [60]. We have separated and expanded this functionality into its own separate component that communicates with the SCIM over SIP. The separation of this functionality is important as keeping an up-to-date view of the context of the end user represents a non-trivial load. Additionally there is the need to interface with multiple sources of information, which may be provided by third parties. The SCIM is part of the critical core network and will play a large role in the routing of messages. It is not appropriate for such an element to interact with unknown entities through multiple different interfaces. The CS performs a distinct, separate function to the rest of the service orchestration and acts a source for the information needed to make contextual decisions. A dedicated component handling this allows it to gather information from a wide range of different sources without impacting the real-time processing of SIP requests.

Functionality

The presence service in the IMS is considered to be a building block on which other services can be based and is standardised as an enabler by the Open Mobile Alliance (OMA). Thus it has been chosen as an example of context that can be used to influence service routing decisions. The “*status*” or “*presence*” of the end user normally indicates the willingness and / or the ability of an end user to communicate with others, and in this framework its use will be extended to also influence which services are triggered. The presence service has been detailed in a previous chapter and as such only the role the CS plays will be discussed here. A more detailed explanation of the IMS presence subsystem can be found in [95]. The simplified process below is followed by the CS for each user for which it needs to maintain context, and can be seen in Figure 4.3

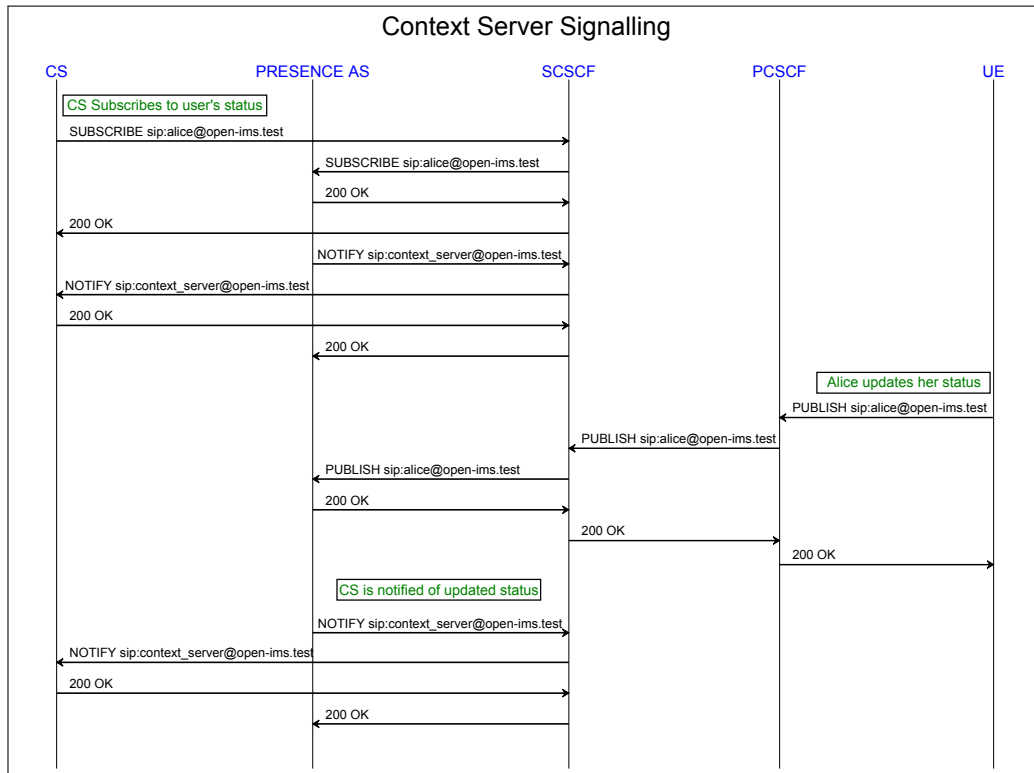


Figure 4.3: Context Server Subscribing to End User Context.

1. The CS subscribes as a watcher to the end users' presence information using a SIP SUBSCRIBE message.
2. The presence server replies with a SIP 200 OK response as well as a SIP NOTIFY message containing the current status of the end user.
3. This SIP NOTIFY message is repeated whenever the end user's status changes.
4. On receipt of a SIP NOTIFY message the CS stores the corresponding information in its own local cache.
5. This information is then transmitted in a single batch to the SCIM after certain criteria are met, such as a set time period expiring. The transmitted information is a condensed summary of the changes to the

contextual information, and is sent to the SCIM in the body of a SIP MESSAGE.

This procedure has been simplified for clarity as it would normally involve the use a resource list and a Resource List Server (RLS) to handle multiple end users subscriptions more efficiently, but it illustrates the process the CS follows.

4.3.3 Service Repository Server (SRS)

The Service Repository Server (SRS) is used to hold information about the available services that are on offer in the network. This information is used for conflict detection and resolution, as well as allowing service monitoring.

Traditionally a subset of this information would be manually programmed into the S-CSCF and other core network elements both in the existing standardised architecture as well as the proposed architectures discussed in the related literature. However, with the proposed framework this information is transmitted automatically to the SRS. This reduces the manual overhead needed to maintain and configure a set of services, as well as allowing the service to update key elements of its own configuration. On the first start of the Application Server (AS) it sends its configuration information to the SRS through the use of a SIP PUBLISH request containing the XML document described above (see Figure 4.1).

This constitutes the service *registering* with the enhancements provided in the proposed framework. Once the service has been registered, it can be referenced and used through out the rest of the proposed enhancements. In addition to this registration process, this mechanism also allows the service to continually report its key metrics to the network. These can include resource utilisation, such as the current CPU utilisation or free memory, to more service specific metrics such as number of users currently being served.

These metrics are contained in the XML document mentioned above, and the same SIP PUBLISH mechanism is used for this as well.

This information would be exposed to the network operator, for example through a web interface, allowing the network operator to view the current list of available services as well as their resource utilisation. An example of the signalling used for this can be seen in Figure 4.4.

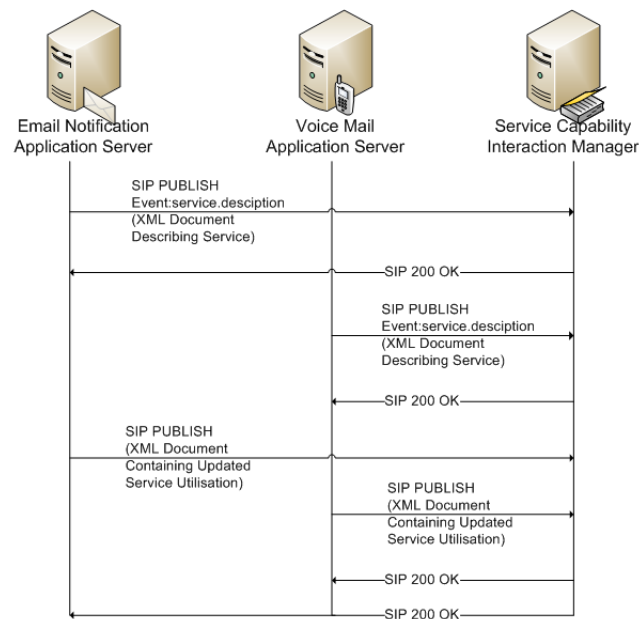


Figure 4.4: Example signalling flow for service monitoring.

4.3.4 User Service Policy Server (USPS)

The User Service Policy Server is the component responsible for interacting with the end user, allowing them to input their desired service execution scenario. It allows them to view available services and to create customised service flows. In other words, it allows them to explicitly detail under which conditions a particular service should be activated.

The interface between the end user and the USPS is based upon the HTTP protocol as this is the most common protocol supported by most end user

devices. The end user browses a webpage that provides information on the available services, together with a simple GUI to drag and drop services and contextual information together to form a particular service activation scenario. To achieve this goal, the USPS needs to have both a list of services as well as a list of conditions that the end user can choose from. The list of services is built from information received from the SRS. The SRS periodically sends a SIP MESSAGE request to the USPS containing a list of any recently updated services together with their description and applicable parameters.

The list of conditions can be supplied in this information for service specific conditions, and is augmented with a list of non service related conditions preconfigured by the network operator such as types of presence supported (*“busy”*, *“offline”* etc) or particular time slots. This extra information is supplied via an XML configuration file.

The USPS is also the first node discussed here to provide conflict detection mechanisms. During the end user’s interaction with the system it examines the choices being made and will not allow the end user to create a service usage scenario that contains either conflicting services or conflicting conditions.

For example, should the end user attempt to choose two different services to be activated that conflict with each other the system will alert the end user to this choice and will not allow the end user to continue creating their usage scenario until the conflict is resolved. Similarly it will prevent the end user creating a scenario with different services to be activated under the same conditions.

In addition, if the end user is creating multiple usage scenarios the USPS will not allow the end user to add another scenario that activates a different set of services under the same conditions. Rather the original scenario that is in conflict must be edited to the end user’s new wishes instead of creating an additional usage scenario. These are some examples of conflict detection mechanisms that can be put in place. The USPS presents the necessary

framework to achieve this, and it is expected that it will be expanded upon in the future.

4.4 Discussion

The previous sections have introduced several new types of functionality or subsystems. These need to be compared to the requirements proposed at the beginning of the chapter to ensure that they have been covered sufficiently. They can be seen on the following page in Table 4.1.

This chapter has proposed a new architecture for enhancing service orchestration in the IMS. The additional elements add flexibility without sacrificing either backwards compatibility or efficiency. The presented architecture introduces information gathering capabilities, the ability to make contextual routing decisions and allows the end user to customise their service usage. It does this by introducing additional elements that can be deployed side by side to the original IMS architecture. This allows for an upgrade path where enhanced services are slowly added to the network. The next chapter details a proof of concept implementation of the architecture which was used to validate the design.

Requirement	Corresponding Functionality
Minimal change to the existing architecture	All architectural changes introduced are backwards compatible, and allow a migration path for existing deployments. No deep changes to the AS servers are required.
Increase flexibility of ATA	The existing functionality is kept. New functionality in terms of the dynamic service orchestration added allows a higher degree of flexibility when triggering services. It is also possible to now use external information in service routing, which increases the flexibility of the architecture.
Allow end user control	The USPS introduces an interface for the end user to control their service triggering for the first time.
Service conflict detection / resolution	There are several additions that improve the handling of service conflicts. Information about different services is gathered and described in such a way that conflicts can be determined. Service conflicts can be determined during the initial configuration of service combinations by the end user or during session activation by the SCIM.
Avoid redundant signalling	No additional redundant signalling is added during the session establishment. Out-of-band signalling is used to convey information about the services and decisions that need to be made, so that session establishment is not effected negatively.

Table 4.1: Requirements versus added functionality.

Chapter 5

Implementation

A working implementation of the architecture proposed in the previous chapter was developed as a proof of concept to show that the presented ideas have merit and are achievable in a realistic manner. This prototype was also used to refine some of the ideas expressed in the previous sections and remove some unseen problems, resulting in a stronger architectural model. This chapter describes the decisions taken while developing the implementation, followed by details for the individual components built for this thesis.

5.1 Implementation considerations

While strictly not related to the theoretical concepts presented in the previous chapter, one must be aware of the real world implications of designing any architecture or software. There are a couple of points to be made regarding this before we examine the implementation of the individual components described in the previous chapter. One of the aims of the work contained in this thesis is to further research in the IMS area and as such it was decided to release the source code of the implementation under a permissive open source licence [106]. This allows the code to be used by as wide a target

audience as possible, allowing researchers to modify the software as needed for their own research as well as allowing the software to use open source libraries.

5.1.1 Platform and Language Choice

A decision needed to be made on the desired platform and suitable programming language to be used for the research contained in this thesis. While the author has had previous experience developing IMS based applications in C for the Linux platform [107–109], it was discovered that there was a lack of suitable IMS based work for the Windows platform, both in terms of server components as well as client components. It was also observed on some popular mailing lists [110] that fellow IMS researchers battled with a fair number of problems relating not to the IMS but the Linux platform for which most of these tools were developed.

As such it was determined that any software developed should be able to run on the Windows platform and this informed the choice of C# as the programming language. Applications written in C# can be run on both Windows and Linux platforms (through the use of Mono [111]), allowing the researcher to use the platform they are most comfortable with. C# was also attractive due to the ease at which one could create professional looking Graphical User Interfaces(GUIs), and during the initial investigation there were also several open source SIP libraries written in C# that could be utilised. Thus it was decided to implement the components in C# targeting Windows and Linux hosts.

However, it must be noted that since this work began there have been new windows based IMS clients developed. The landscape has changed significantly since the work was started, and there are now freely available windows compatible IMS clients, indicating that there was such a demand [112, 113].

5.2 SIP Library

The success of the work contained in this chapter is heavily influenced by the SIP library, as each component utilises SIP signalling. However, it soon became apparent that the available C# SIP libraries were not conducive to IMS control signalling and research.

While pre-existing C# SIP libraries existed (most notably Lumisoft's SIP library [114]) these proved difficult to work with. They lacked the necessary extensions to SIP that the IMS uses (e.g. the Service-Route header [115] amongst others) as well as hindered research by placing constraints on the SIP message contents. For example, the Lumisoft SIP library automatically generated some SIP headers that were then not possible to change through the library mechanisms, and it did not support proper logging, either of the parsed SIP messages or the raw data. Thus it was decided to create a new C# SIP library, where the emphasis could be placed on research friendly design decisions. This library controls all the SIP signalling in each of the components discussed further on.

The key elements that make it research friendly are:

- Event driven architecture with hooks at multiple levels through the SIP stack either for logging purposes or for actions to be taken
- No constraints placed on interacting with or editing SIP messages
- Functions to automatically generate the appropriate values for most of a message's headers (but still allow the editing of these headers).

The SIP library is compiled to a DLL file which is then added to the other components. This allows the SIP library to be updated and redeployed to all components fairly easily, and keeps the actual SIP signalling well separated from the other functionality. At the heart of the SIP library is an abstract class called SIPApp which exposes abstract functions such as

`ReceivedRequest` or `ReceivedResponse` which are overridden by the specific application or component using the SIP library. This allows the other software components to dictate what happens on such events as receiving a SIP request or the creation of a dialog, without having to worry about the internal state management of the SIP transactions. These abstract placeholder functions can be seen in Figure 5.1.

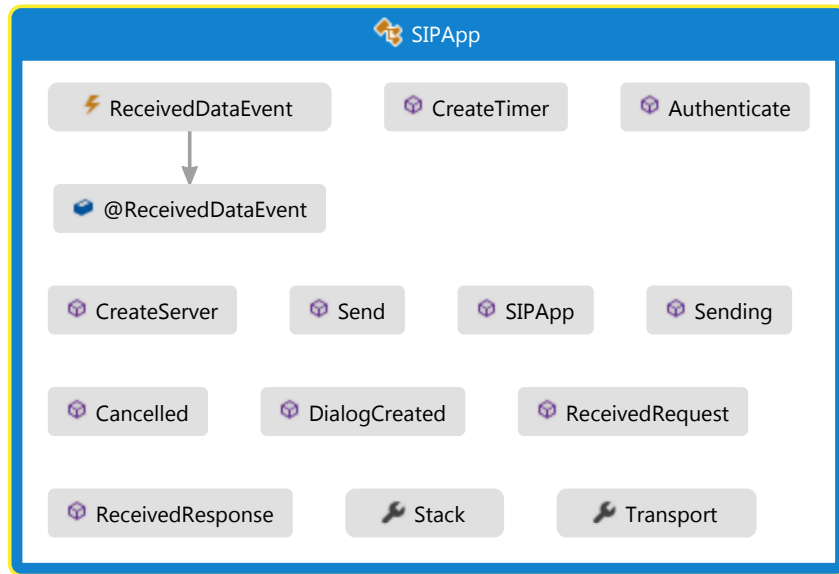


Figure 5.1: Abstract methods and events of the SIPApp Class.

5.3 Client

As this thesis deals with allowing the end user to customise and enhance their service usage through the end user's context, any implementation to validate the concepts needs to contain some form of end user IMS client. Due to the lack of available IMS clients for Windows that were appropriate for our research needs, we found it necessary to fully implement an IMS client to properly test various scenarios in a realistic manner. This allowed us the flexibility needed to rapidly make changes to the client's source or to determine what was happening in the SIP signalling. As such the client was

designed to be research friendly. This means that it has full logging of all functionality, including SIP messages as well as raw network traffic, and major functionality is provided by distinct modules. An example of the logging capabilities are shown in Figure 5.2. Each module is linked together through the use of messages or events, allowing a high level of separation between different sets of functionality. The separation allows the easy replacement or extension of existing functionality. A high level overview has been provided in Figure 5.3, in which some of the modules can be seen.

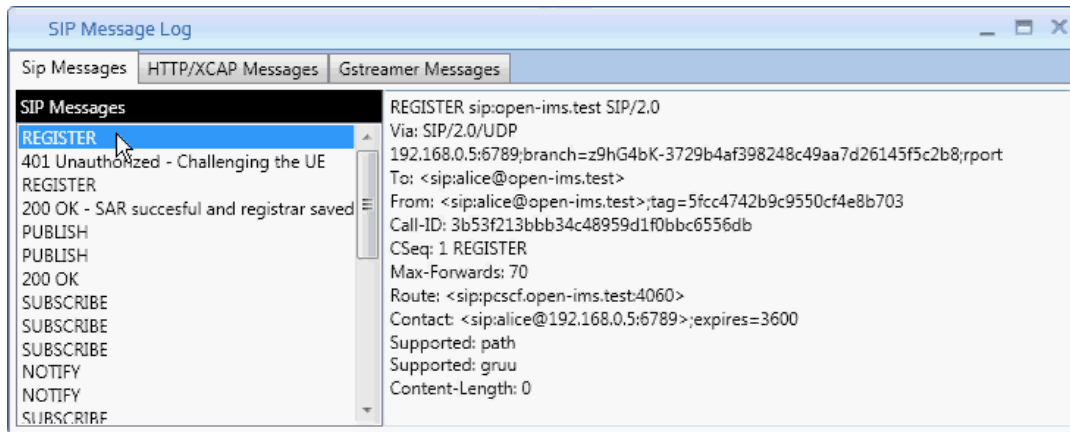


Figure 5.2: Debug Log showing SIP signalling.

5.3.1 Signalling

As described above, a SIP library was developed which performs the basic SIP signalling. However, in addition to the SIP library component there are also specific modules for handling instant messaging and voice/video calls. These wrap the functions provided by the library into easy to use components which can pass messages between the different sets of functionality, without having to know the details of the actual SIP signalling.

For example, the instant messaging module exposes a event with the contents of any received SIP message which can be listened to by another module (for example to update the GUI, log the message, or to trigger a call etc).

This allows other modules to integrate different sets of functionality without having to worry about the SIP signalling used.

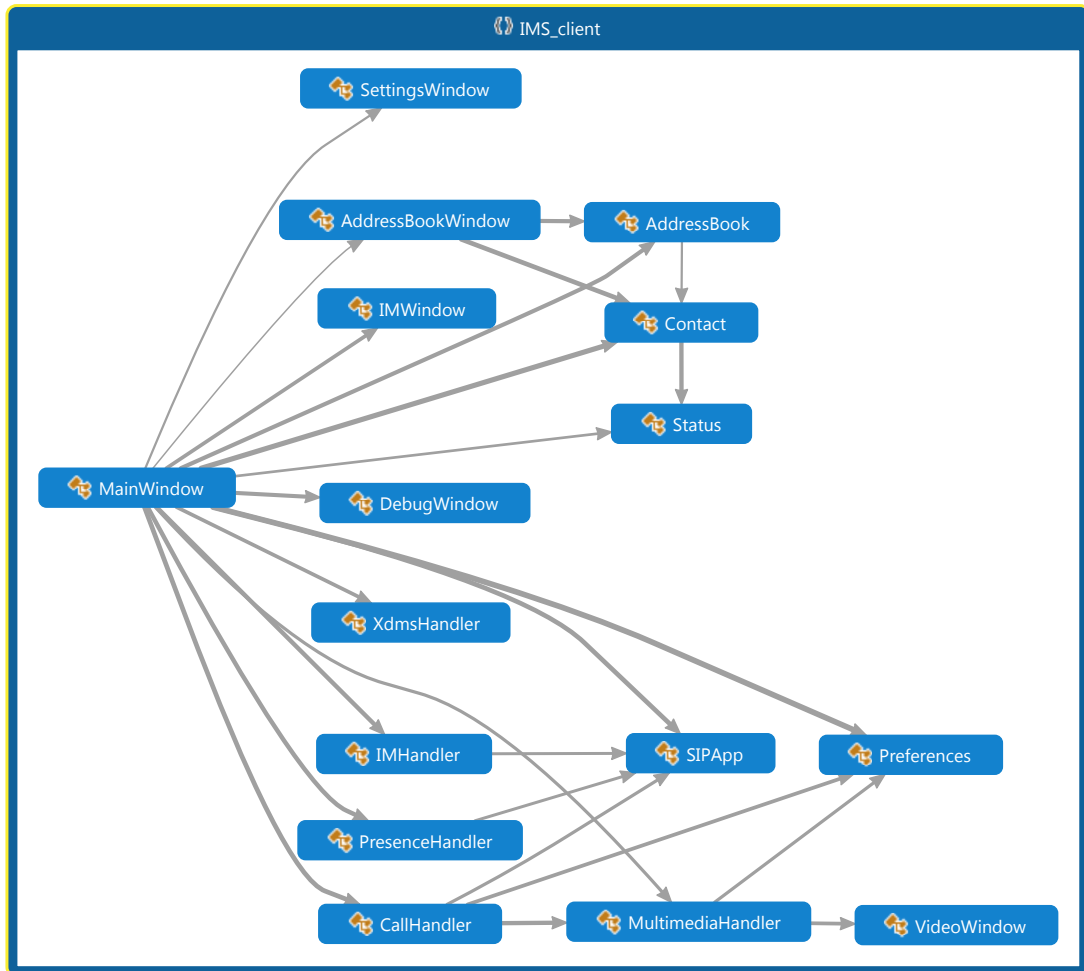


Figure 5.3: Client modules.

5.3.2 Registration

The first stage needed for any IMS client is to register with the IMS network. As such the client can register and de-register using SIP REGISTER requests with the necessary IMS extensions.

5.3.3 User context

The proposed work revolves around the concept of user context, and as such it is necessary to have a client that can provide some form of context. The context system chosen for the implementation was the standard IMS presence mechanisms. This is handled by the presence module, which contains functionality to handle both SIP SUBSCRIBE requests as well as SIP NOTIFY requests to publish the end user's own status as well as receive notifications of other users' status changes. These changes are also exposed by the module using the event system so that other modules may perform their own actions based on status changes.

5.3.4 Multimedia

The multimedia used by the client is provided through the use of the GStreamer library [116]. This provides the necessary codecs and surrounding functionality for both voice and video calls, as well as various supplementary sounds such as ringtones. The GStreamer library was chosen as it supports a wide range of different codecs through its plugin system, as well as being available for multiple programming languages (C# included) on multiple platforms (including Linux and Windows platforms). Multimedia events are also exposed through the event system.

5.3.5 Services

The client supports both video and audio calls with a range of codecs that can be negotiated through the use of SDP. Instant messaging with typing notifications has also been implemented. The client supports the XML Configuration Access Protocol (XCAP), allowing it to communicate with an XML Document Management Server (XDMS). This functionality is used to operate

a network-stored address book which provides access to a uniform, synchronised contact list that is available from different computers or terminals.

5.4 Framework Components

The framework components utilise the same SIP library that was developed for the client implementation. However, they do not expose GUIs to the end user, and are specifically developed as console applications for either Windows or Linux.

5.4.1 Context Server (CS)

The implemented functionality for the CS consisted of the necessary SIP signalling to watch an end user's presence and communicate this to the SCIM. The CS subscribes to the watched users on start-up. This is done through the use of SIP SUBSCRIBE messages, and it listens for any incoming SIP NOTIFY messages. These messages contain updates to the end user's status. It builds a local cache of the current status of the end users and sends this to the SCIM after a fixed timer interval by sending a SIP MESSAGE request.

5.4.2 Service Repository Server (SRS)

The SRS contains both a SIP processing component as well as an HTTP interface. It receives SIP PUBLISH messages from the AS which contain the XML describing the service offered by the AS. As this was a proof of concept implementation, the data store used for saving this information was chosen for flexibility and ease of use as opposed to performance. As the SRS manipulates information about services which each have a unique key associated with them, a key based data store was chosen. The received service description is stored in a *B+tree* (a variant on the traditional B-tree) which

allows efficient insertion, searching and retrieval operations with disk backed persistence, allowing easy modification of saved state as well as reducing the infrastructure requirements for running this component. The SRS updates a local cache of changed services on receipt of any SIP PUBLISH requests that contain an updated service description. After a fixed interval has expired the SRS checks this cache for any new entries, and if any modified services are found they are sent to the SCIM in a SIP MESSAGE. The SRS also exposes the current service monitoring information through an HTTP interface. This can be used to see the resource utilisation (CPU load, Available Memory etc.) of any service as reported to the SRS.

5.4.3 User Service Policy Server (USPS)

The USPS is implemented as a traditional web application. It has gone through several different prototypes during development, including a traditional ASP .NET web page and a version implemented entirely in JavaScript (using Node.js for the server side processing). The ASP .NET version integrated easily with the other elements as all components were written in C#. However the front-end interface lacked the necessary functionality we desired, namely the easy manipulation of graphs with distinct nodes.

The JavaScript version allowed us to use the powerful Data Driven Documents (D3) framework to provide the desired front-end [117], but difficulties were encountered syncing information between the different elements because of the different formats used on each side. There was also a need for a new JavaScript compatible SIP library.

The two implementations were combined with the C# ASP .NET handling the back-end and all user interfaces completely written in HTML, CSS and JavaScript. XML web services are used to update and retrieve information dynamically depending on what the end user is currently doing. This allows the back-end and front-end to be very loosely coupled, allowing each section to be implemented as desired.

The USPS receives the list of available services from the SRS through a SIP MESSAGE request and stores these in a SQL database. Similarly to the implementation of the SRS a file backed data store was used. As the data to be stored was more complex, a SQL database was chosen. However, we still wanted flexibility and ease of use as before. Hence SQLITE, which is a flat file database, was chosen. This is a self-contained transactional SQL database that uses files to store its contents. This lowers the overhead required to deploy the element as a separate SQL server setup is not needed. The list of services with their description is available for the end user to view through a web page, allowing them to learn about new services as well as see how they could be integrated together.

The main function of the USPS is to allow the end user to customise their service usage. This was done by allowing the end user to construct a flowchart or graph indicating how they would like their services to be used. Each step was accompanied by a choice of different options that the end user could choose from, and this can be seen in Figure 5.4.

Once the end user had finished customising their service flow it was examined for any potential conflicts. The implementation allows for this to be extended by adding new rules as functions to be called on the service flow. Currently there are two rudimentary conflict detection algorithms implemented. These do not allow the end user to try to activate two services without the necessary distinct conditions as well as disallowing different values for the same condition to be used in the service flow. This is also where the service description is examined to check that the services chosen do not conflict based on the information stored in their service description (SIP headers read / written etc). If a conflict is detected the end user is alerted and allowed to change the service usage scenario to avoid the conflict. Once the end user had finished customising their service flow, it is saved to the database. After a fixed interval had expired, all updated service flows were communicated to the SCIM through a SIP MESSAGE request. An example of a customised scenario that has been created by the end user can be seen in Figure 5.5.

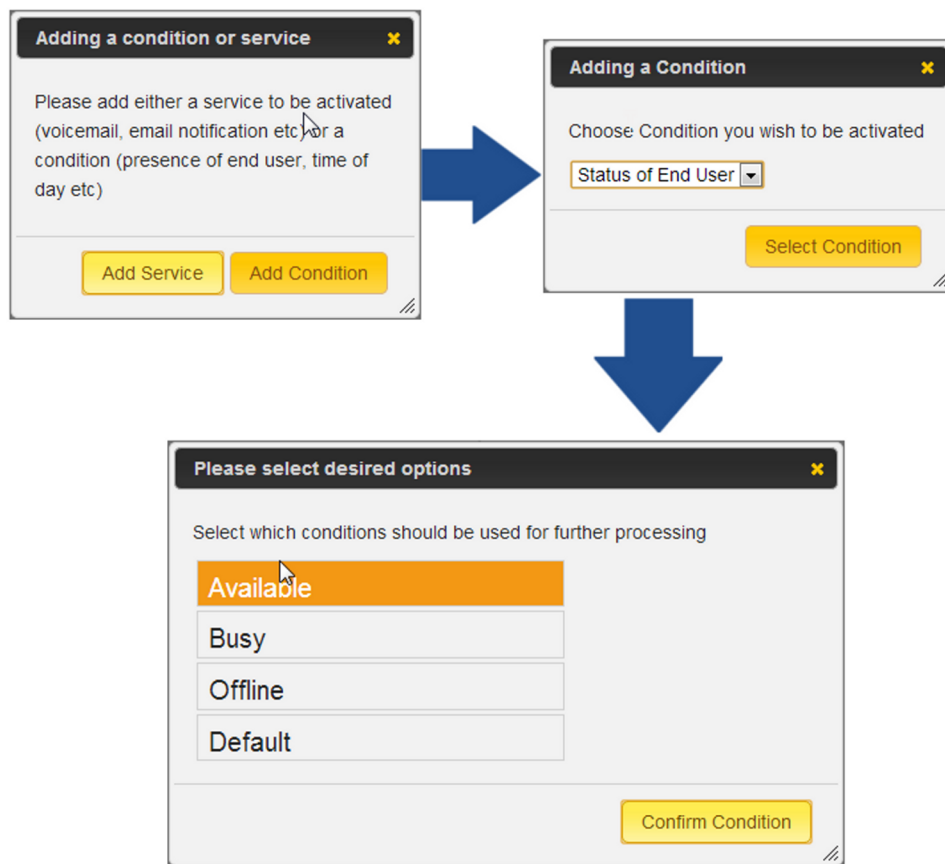


Figure 5.4: Screen shot showing options presented to end user.

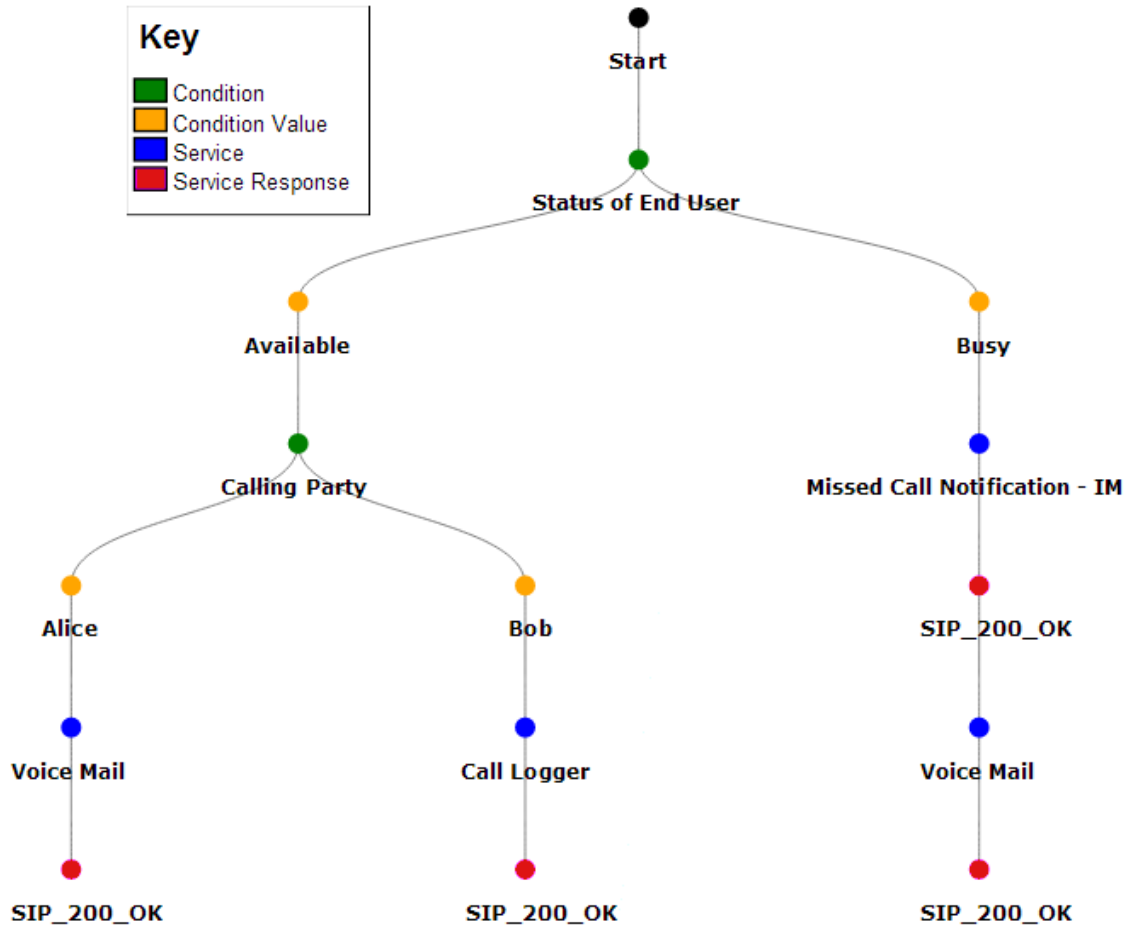


Figure 5.5: Service flow scenario customised by end user.

The proposed conflict detection algorithm is depicted in Algorithm 5.1 on the following page. It displays a warning message when activating different services that rely on the value of the same SIP header, and disallows activating services that overwrite the same SIP headers. It also detects when different services are requested to be activated under the same conditions, i.e. at the same time without any ordering specified.

While it is not a very complicated algorithm, the proposed implementation provides a framework for testing such algorithms, and it is expected that this algorithm will be improved upon in future work. Here it serves as an proof of concept for testing the overall architecture.

Algorithm 5.1 Service conflict detection algorithm

```
1: USPS receives new service flow from end user via HTTP interface.  
   Transitions between stages are based on SIP messages or contextual  
   information.  
2: if the first stage is a service activation stage then  
3:   if any existing Service Flow starts with the same service stage then  
4:     A hard conflict has been detected;  
5:     Reject this service flow;  
6:     STOP;  
7: else if the first stage is a contextual condition stage then  
8:   Build a list of conditions to match until the first service is  
9:   activated for the newservice flow;  
10:  Retrieve existing flows that start with a condition stage;  
11:  for all configured service flows in the list of existing service flows do  
12:    Build list of conditions until first service is activated  
13:    if this list or condition chain matches then  
14:      A hard conflict has been detected;  
15:      Reject this service flow;  
16:      STOP;  
17: for all services in new service flow do  
18:   Retrieve list of all SIP headers READ  
19:   if header exists in both then  
20:     Soft Conflict Detected;  
21:     Warn about service conflict;  
22:   Retrieve list of all SIP headers WRITTEN  
23:   if header exists in both then  
24:     Hard Conflict Detected;  
25:     Reject this service flow;  
26:     STOP;  
27: for all conditions in new service flow do  
28:   if same condition listed more than once then  
29:     if value differs then  
30:       A hard conflict has been detected;  
31:       Reject this service flow;  
32:       STOP;
```

5.4.4 Service Capability Interaction Manager (SCIM)

The SCIM is the central routing control node where the next hop or destination is identified for a particular message that it has received. The algorithm for determining how a message is routed can be seen in algorithm 5.2.

The SCIM uses the Call-ID as a starting point to identify messages. According to the SIP RFC it should be a globally unique identifier over time and space and each UA must have the means to guarantee that these IDs will not be generated by any other UA. The current context information that has been reported by the CS is kept in a local cache to allow contextual information to play a role in the routing of the SIP messages.

The SCIM maintains a list of the customised service flows that have been created by the user and received from the USPS. It also maintains a list of any currently executing service flows. The active service flows are stored in a data structure that holds the original request, the last request seen, the last response seen as well as the last active stage of the service flow. These help track state as well as allowing the SCIM to continue routing the service flow on receipt of any further responses or requests. This information can also be used to compare requests to ensure that they are not modified in such a way to cause conflict. If such a conflict were discovered, the SCIM can control the further routing of the SIP request to ensure an undetermined state is avoided. Each step of the service flow is represented as a node in a tree with both a global ID and a local instance id. The global ID is used to look up information about the particular condition or service it represents, while the local instance ID represents a particular block in a specific service flow. This tree is consulted when a message is received to determine the next destination for a message. Only certain blocks have a destination URI and the algorithm parses the tree until it either finds a destination URI, a condition that is not true or the end of the tree. If the destination URI is not found before the tree parsing function is completed the message is routed according to normal SIP rules.

Algorithm 5.2 SCIM Request Routing algorithm

```
1: The SCIM receives a message, either a SIP REQUEST or SIP RESPONSE
2: if the message is a SIP REQUEST then
3:   if the Call-ID exists in its local cache then
4:     Retrieve the associated service flow;
5:     Determine the current point in the service flow;
6:     Pass to the routing algorithm with the service flow starting
7:     from the current point;
8:   else if the Call-ID does not exist in its local cache then
9:     Retrieve the list of service flows for the recipient;
10:    for all service flows in the retrieved list do
11:      Check the REQUEST against the first stage of the
12:      configured service flow;
13:      if the first stage of the configured service flow is a condition then
14:        Check user's current cached value for condition;
15:        if the two conditions match then
16:          Begin the routing algorithm with the next stage in
17:          the configured service flow;
18:        else
19:          Route the message according to default SIP routing rules;
20:          STOP;
21:      else if the first stage of the configured service flow is a service then
22:        Save the REQUEST information together with the
23:        activated service flow;
24:        Change the SIP TO header of the request to reflect this service
25:        (new destination);
26:        Route the SIP request to the new service;
27:        STOP;
28:  else if the message is a SIP RESPONSE then
29:    if the Call-ID exists in its local cache then
30:      Retrieve associated service flow and last matching request;
31:      Determine point in service flow;
32:      if received response matches the service flow's response value then
33:        begin the routing algorithm with the last request and the
34:        next service stage
35:      else
36:        Route response according to default SIP routing rules
37:        STOP;
38:  else if Call-ID does not exist in SCIM cache then
39:    Route SIP response according to default SIP routing rules;
40:    STOP;
```

The SCIM caches the received contextual information or the user created service flows locally, updating them as needed from new requests received from the framework. The algorithm used to route any incoming SIP requests from the end user is shown in Algorithm 5.2 on the previous page.

5.5 Discussion

This chapter has discussed the implementation details for a prototype of the proposed architecture. Whilst the components are not intended to be highly performant due to the exploratory nature of their implementation, they do successfully extend the IMS architecture in the required manner, allowing a much wider range of functionality than before. As the proposed architecture consists of additions to the standardised IMS architecture, a fully functional IMS test bed was needed to ensure the correct operation of the proposed elements.

Chapter 6

Evaluation Platform

The previous chapter described the actual prototype implemented as part of this thesis. However, the prototype does not exist without a supporting framework. It is part of a larger framework which has been standardised by the 3GPP. Thus to test the implementation it was necessary to construct a test-bed consisting of these additional components. This chapter describes the test-bed used to aid in the understanding of the test results in the following chapter as well as serving as guidance should other researchers wish to evaluate similar works or construct their own test-beds.

6.1 Test-bed environment

The proposed architecture introduces new elements in the signalling path of the core IMS network. As this influences the session establishment delay, it was decided to run each element of the IMS core network separately. We wished to avoid artificially limiting the delay between the components by co-locating them on the same machine. This would have resulted in serving network requests over the local loop back interface, greatly reducing the latency of these requests.

However, acquiring, maintaining and deploying eight or more servers in a consistent, repeatable fashion is not an easy task, nor is it a particularly efficient use of resources. Initially the implementation was designed and developed on a physical hardware cluster, details of which can be found in a previous paper [118]. This provided a set, static amount of computing resources. It did not allow for flexibility in ensuring that any one component was not acting as a bottle neck and unduly influencing the delay characteristics of the session establishment.

For example, during testing it was discovered that particular components were experiencing high load, with or without the proposed changes. As such there was a need for changing the computing resources available to individual components to ensure that any performance impacts were not caused by unrelated artifacts but reflected the actual change introduced by the new architecture.

A second factor that went into this consideration was the fact that this work was focused solely on the core network. The changes implemented by the new architecture do not require new SIP signalling from the client equipment, and as such no new SIP control messages flow over the access networks. The standard SIP signalling between the client equipment and the P-CSCF has been investigated elsewhere [73], and lies outside the desired functionality of the test-bed. Thus there was not a specific need to evaluate performance characteristics of different access networks, and having a isolated test environment running in the cloud was determined to be acceptable.

It was decided that a test-bed that provides elastic computing resources would be suitable for the desired testing environment, and hence Amazon's Elastic Compute Cloud (EC2) was the service chosen to host the test-bed. Generally there is a large movement towards cloud based provisioning for non-telecommunication based IP workloads. More specifically for this area there is literature evaluating IMS based systems integrating or running on public cloud infrastructure [119–121]. This environment allowed us to easily reconfigure components as needed when running the test suite.

6.1.1 Supporting infrastructure

The test bed comprised of virtual servers used for the IMS components as well as additional virtual servers to provide the necessary background infrastructure. While not necessarily part of the core infrastructure being evaluated, three additional virtual servers were necessary to complete the test-bed. The first virtual server hosted the DNS server that was used by each component to perform name lookups and find the other components. It also acted as a base for running various scripts and automation across the test-bed.

The second virtual server served as a VPN endpoint. This provided a secure entrance point to the test-bed, allowing non public IPs to be used amongst the components as well as allowing less restrictive firewall policies to be deployed. The entire test-bed was firewalled off from the general internet, with it only being accessible through the VPN provided by this virtual server. This also allowed other machines outside the testbed to be connected onto the testbed local network via the VPN, for retrieving results from the test runs or uploading new software components etc.

The last virtual server used was the Test Manager virtual server. This server acted as both the traffic generator and the co-ordinator of the tests that were executed across the test bed. This VM ran one instance of the IMS Bench SIPp manager configuration to control the test run and gather the monitoring results, as well as running one instance of the SIPp load generator to send traffic towards the IMS network running in the test-bed.

6.1.2 A note on performance

As stated previously, performance was not the key metric under evaluation here. However, it must be noted that while public cloud providers provide great flexibility in configuring compute resources, several caveats have to be discussed when running tests on such platforms. They allow a large range of technologies and computing capacity to be provisioned at an incredibly

low price point (compared to having to purchase and install such hardware), but introduce some uncertainty in the performance characteristics of any particular environment. This has to be taken into account when running tests on such platforms.

For example, the actual server instances may not be co-located on the same host machine, nor in the same rack or even the same data-centre. This can skew results should one virtual server be located in such a way that the latency between one host and the other hosts is greatly different than between the other hosts. This was encountered during our testing, and initially gave higher delay measurements than originally expected. The delay from one of the virtual machines was approximately 5 milliseconds higher during one of the test runs, resulting in the expected graph but shifted upwards by approximately 10 milliseconds.

To account for this, continual measurements of the latency between the test manager and the other hosts was carried out through the use of smoke ping [122]. This ensured that testing was only conducted when the baseline measurements were acceptable, and prevented transient delays from affecting the results.

Monitoring of the various other resources such as CPU or memory was also conducted during the tests to ensure that there were adequate resources to gain meaningful results. Occasionally a VM was found to be under performing in terms of poor CPU or network performance; a common occurrence in elastic cloud providers [123, 124]. Stopping such a problematic instance and launching a new one after a short pause often resolved this issue, due to the large number of physical machines in the cloud provider’s data centres. Relaunching the instance (as opposed to simply rebooting) will very often place the instance on a new physical machine.

Both physical factors (such as type of hardware or physical location) as well as non physical factors (such as workload of co-located VMs, the so-called “Noisy Neighbour” impact) can lead to irregular performance, and running

the VM on a new physical machine can alleviate these types of performance problems.

6.2 Test-bed components

While the IMS is based heavily on well known protocols such as SIP and Diameter, there are also several extensions to these protocols which are required for the IMS network. This means that one can not simply use SIP proxies or routers without further customisation for the Call Session Control Functions (CSCFs) even though they share a large degree of functionality with SIP proxies. Thus there is a need for specialised implementations of the CSCFs.

Here we chose to use the Fokus Open Source IMS Core (OSIMS) [125–127]. This core implementation provides the necessary authentication, authorisation and default SIP routing mechanisms to create a working IMS network. It also provides the necessary mechanisms to store user and service profile data in accordance with the requirements laid out by the 3GPP functional specifications, allowing end users to register with the network and utilise any services which have been added to the test-bed. It provides the CSCFs and the Home Subscriber Server (HSS), which together provide all the core functionality needed in an IMS network. The OSIMS has been used for many years as a reference implementation for a System under Test (SuT) by various groups, including Workgroup 6 of ETSI TISPAN who have standardised the benchmarking procedure to be used for NGNs [128]. This, together with its open source nature, made it an ideal choice for this project.

6.2.1 Call Session Control Functions

The CSCFs can be broken down into three main different types, namely the Serving CSCF (S-CSCF), the Proxy CSCF (P-CSCF) and the Interrogating

CSCF (I-CSCF). The functionality of these components has been discussed previously. As mentioned above a large portion of the CSCF functionality is similar to the functionality provided by a SIP proxy, e.g. forwarding on SIP messages.

However, functionality is also needed in terms of acting as a registra and as a back-to-back user agent (B2B-UA). The CSCFs also need to be able to support Diameter and the various authentication mechanisms specified by the 3GPP. The CSCFs provided by the Fokus Open Source IMS Core were based upon the SIP Express Router (SER) which had previously been well regarded and developed by the same institute.

This has lead to a scalable, high performance framework that can be easily extended as these components are provided as open source software to the research community free of charge. Together these elements handle the core IMS functions such as user registration and controlling the signalling between all IMS elements, and are written in C for best performance.

6.2.2 Home Subscriber Server

In comparison to the CSCFs, the HSS is written in Java, and configured through the use of a web interface. It is a store of information related to the end user and various services, and interacts with the CSCFs using Diameter over the *Cx* interface. It also supports communicating with ASs via the *Sh* interface [129].

The web front-end allows for easy administration of user profiles and the initial Filter Criteria, including setting up ASs and configuring SPTs etc. It can be seen in Figure 6.1.




FHoSS - The FOKUS Home Subscriber Server (Rel. 7)

[HOME](#)
[USER IDENTITIES](#)
[SERVICES](#)
[NETWORK CONFIGURATION](#)
[STATISTICS](#)
[help](#)

User Identities

- IMS Subscription
Search
Create
- Private Identity
Search
Create
- Public User Identity
Search
Create

Private User Identity -IMPI-

ID	4
Identity*	alice@open-ims.test
Secret Key*	alice
Authentication Schemes*	
Digest-AKAv1 (3GPP)	☒
Digest-AKAv2 (3GPP)	☒
Digest-MD5 (FOKUS)	☒
Digest (CableLabs)	☒
SIP Digest (3GPP)	☐
HTTP Digest (ETSI)	☒

Associate an IMSU

IMSU Identity		Add/Change
---------------	----------------------	------------

Associated IMSU

ID	IMSU Identity	Delete
1	alice	☐

Create & Bind new IMPU +

Associate IMPU(s)

IMPU Identity		Add
---------------	----------------------	-----

Warning: The current IMPI will be associated with all the corresponding IMPUs (within the same implicit-set)!

Figure 6.1: Screenshot showing web interface to FHoSS.

6.2.3 Presence Subsystem

In addition to the core network, a presence subsystem was needed to provide example contextual information for testing the theories involved in this work. One such concept was the routing of SIP sessions depending on the end user's context. Here context was chosen to be status or presence, and the presence subsystem provided the necessary mechanisms to provide this context to the framework. These mechanisms included the ability to manage the subscriptions of various different entities as well as the necessary watcher authentication and access control functions.

OpenXCAP is an open source XCAP server that allows SIP SIMPLE clients to manage buddy lists and to create the necessary policies that allow subscriptions to presence or other types of events. OpenXCAP is written in Python for the Linux platform, which allowed it to be integrated easily with the rest of the test-bed.

However, since all communication with OpenXCAP is through standardised protocols, it could be replaced seamlessly with an alternative product, even

one targeted at the windows platform. Clients use XCAP (through HTTP) to create / modify / delete their buddy lists and access rules which are stored in a MySQL database by OpenXCAP.

Another SIP presence server is needed to act on this information, and we used the Open SIP Server (OpenSIPS) for this functionality. OpenSIPS is written in C and was based upon the same SER project that formed the core of the CSCFs used in the test-bed. OpenSIPS handles the actual SIP SUBSCRIBE and NOTIFY requests, with OpenXCAP handling the modification of authorisation information according to requests made by the client through XCAP. OpenXCAP, together with an OpenSIPS server, provided all the necessary functionality for the presence framework. This system can be seen in Figure 6.2.

6.3 Test-bed functionality

6.3.1 Registration

It is usually required for the end user to register with the network before they can send and receive messages through the IMS network, as well as utilise services provided by the Application Servers. The handling of register requests in the core network was supplied by the OSIMS project and was not modified. An example message of the developed client registering with the network in the test-bed is supplied in Figure 6.3.

6.3.2 Standardised Application Triggering Architecture

Once registered on the network, the end user can use various services on offer, including the presence subsystem. OpenXCAP and OpenSIPS provide the ability to subscribe to changes in end user presence information, allowing the receipt of notifications and the publication of status information as well.

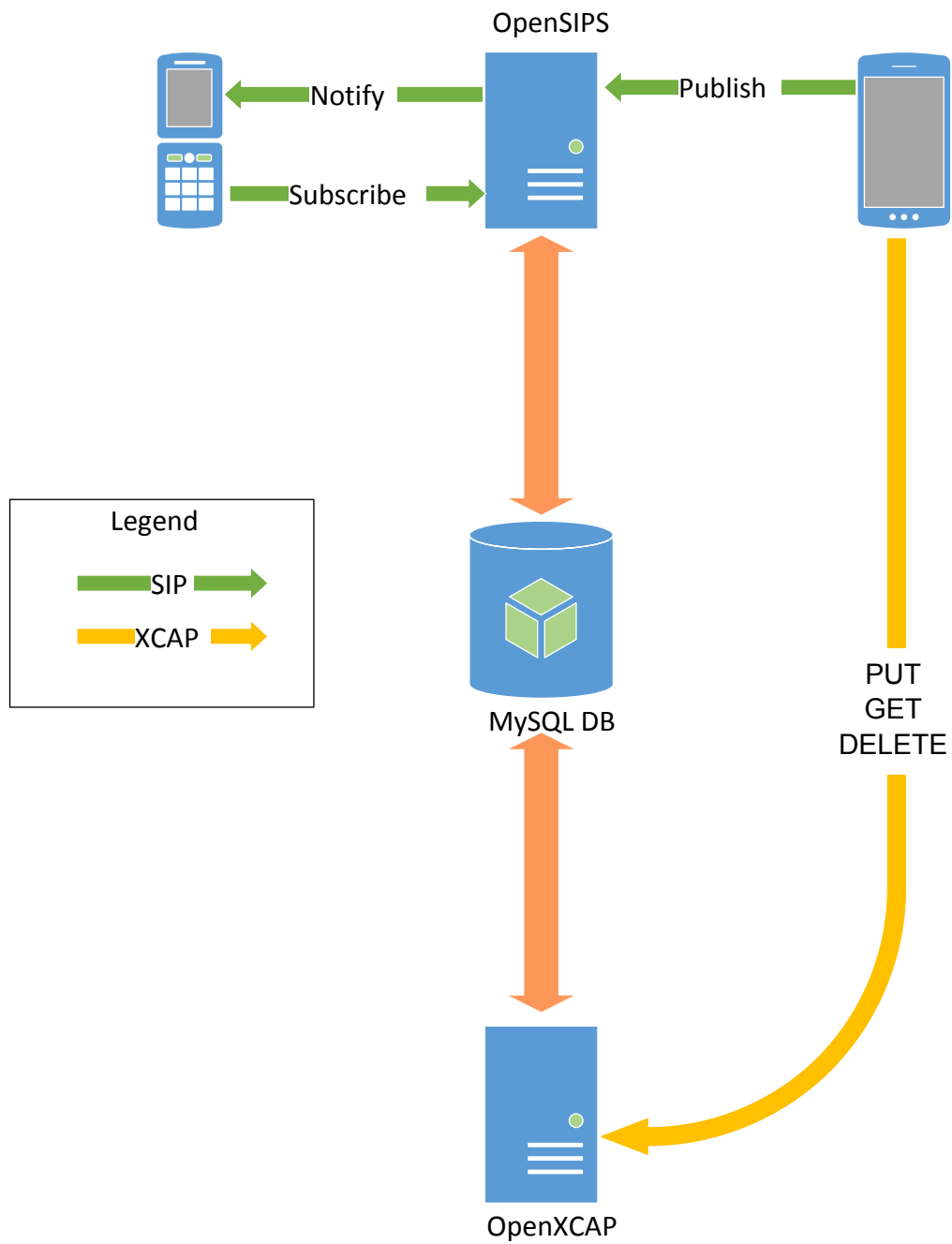


Figure 6.2: OpenXCAP and OpenSIPS based presence system.

```
REGISTER sip:open-ims.test SIP/2.0
To: <sip:alice@open-ims.test>
From: <sip:alice@open-ims.test> ;tag=827845320598027968
CSeq: 2 REGISTER
Call-ID: 1266446281@192.168.20.20
Max-Forwards: 70
Via: SIP/2.0/UDP 192.168.20.20:6789;branch=z9hG4bKfHx8fEZhbHNl
Contact: <sip:alice@192.168.20.20:6789>
Authorization: Digest username="alice@open-ims.test", realm="open-ims.test",
nonce="e5abe83eae404b90659ee9b19e2c14c7", algorithm=MD5, uri="sip:open-ims.test",
response="e7335973ae73c84194904d421d9917d1"
Expires: 3600
User-Agent: SIPLIB
Route: <sip:pcscf.open-ims.test:4060>
Content-Length: 0
```

Figure 6.3: Client Registering with Fokus OSIMS.

However, the way that these messages are routed between OpenSIPS and the end user are controlled by the core network, through functionality provided by the OSIMS. The necessary iFC are configured through the FHoSS web interface, and an example of such a configuration for the presence AS can be seen in Figure 6.4.

To test the architecture described in this work, it was necessary to add several iFC to the existing standard setup. This allowed the forwarding of the SIP requests to the SCIM, allowing it to participate in the routing decisions as opposed to the S-CSCF. This was done through the creation of an iFC with a very broad Trigger point (TP) that matches all the relevant requests.

The architecture allows for different TPs to be configured to route different message sets or requests to the SCIM depending on the network operator's desired scenario. For example, it may only redirect messages coming from a certain class of users or certain types of requests. Here it was sufficient to only redirect SIP INVITE and SIP MESSAGE requests to the SCIM for the purposes of this test-bed, as can be seen in the TP configuration shown in Figure 6.5.



Fraunhofer
Institute for Open
Communication Systems



FOKUS
testbeds

FHOSS - The FOKUS Home Subscriber Server (Rel. 7)

HOME
USER IDENTITIES
SERVICES
NETWORK CONFIGURATION
STATISTICS

help

Services

- Service Profiles

Search

Create
- Application Servers

Search

Create
- Trigger Points

Search

Create
- Initial Filter Criteria

Search

Create
- Shared IFC Sets

Search

Create
- DSAI

Search

Create

Application Server™ -AS-

ID

1

Name*

default_as

Server Name*

sip:192.168.20.248:5065

Diameter FQDN*

presence.open-ims.test

Default Handling*

Session - Continued ▼

Service Info

Rep-Data Limit

1024

Sh Interface - Permissions

Permission for	UDR	PUR	SNR
Allowed Request	☒	☒	☒
Repository-Data	☐	☐	☐
IMPU	☐	☐	☐
IMS User State	☐	☐	☐
S-CSCF Name	☐	☐	☐
IFC	☐	☐	☐
Location	☐	☐	☐
User-State	☐	☐	☐
Charging-Info	☐	☐	☐
MS-ISDN	☐	☐	☐
PSI Activation	☐	☐	☐
DSAI	☐	☐	☐
Aliases Rep Data	☐	☐	☐

Mandatory fields were marked with "*"

Save

Refresh

Delete

Attach IFC(s)

Select IFC...

Attach

Trigger Point -TP-

ID5

NameSCIMTP

Condition Type CNFConjunctive Normal Form

Mandatory fields were marked with "*"

Save

Refresh

Delete

Attach IFC

Select IFC...

Attach

List of attached IFCs

ID	IFC Name	Detach
5	SCIMIFC	☐

Add SPTs to Trigger Point

Not☒

SIP Method

PUBLISH

OR

Request-URI

AND

Not☒

SIP Method

SUBSCRIBE

OR

Request-URI

AND

Not☒

SIP Method

REGISTER

OR

Request-URI

AND

Not☐

SIP Method

INVITE

OR

Request-URI

AND

Not☐

SIP Method

MESSAGE

OR

Request-URI

AND

Delete

Delete

Delete

Delete

Delete

Reg☐ ReReg☐ DeReg☐

6.4 User equipment

No IMS test-bed would be complete without some form of client software that can represent the end user's terminal or device. For this project a custom IMS client was built for IMS specific functionality, and a standard web browser was used for the HTTP / web interface to the system. The UE software was run both on a physical desktop machine as well as on a virtual server at different stages of development and testing.

6.4.1 IMS client

During the testing phase of this project multiple instances of the IMS client were run as needed, depending on the parameters of the tests. The following functionality was required and the IMS client needed to be able to:

- Register with the IMS core network
- Subscribe, publish and display changes in presence information
- Send and receive instant messages
- Make and receive audio / video calls

The IMS client developed for this thesis could register with the IMS network using SIP REGISTER requests, handle presence information using SIP PUBLISH / SUBSCRIBE requests as well as display the content of SIP NOTIFY requests.

It could also handle IM sessions using SIP MESSAGE requests as well as make and receive audio / video calls through SIP INVITE requests and the use of the GStreamer multimedia framework [116]. An example of this functionality can be seen in Figure 6.6. Each client has registered with the core network, received status updates about their online contacts and are exchanging instant messages with typing notification support enabled.

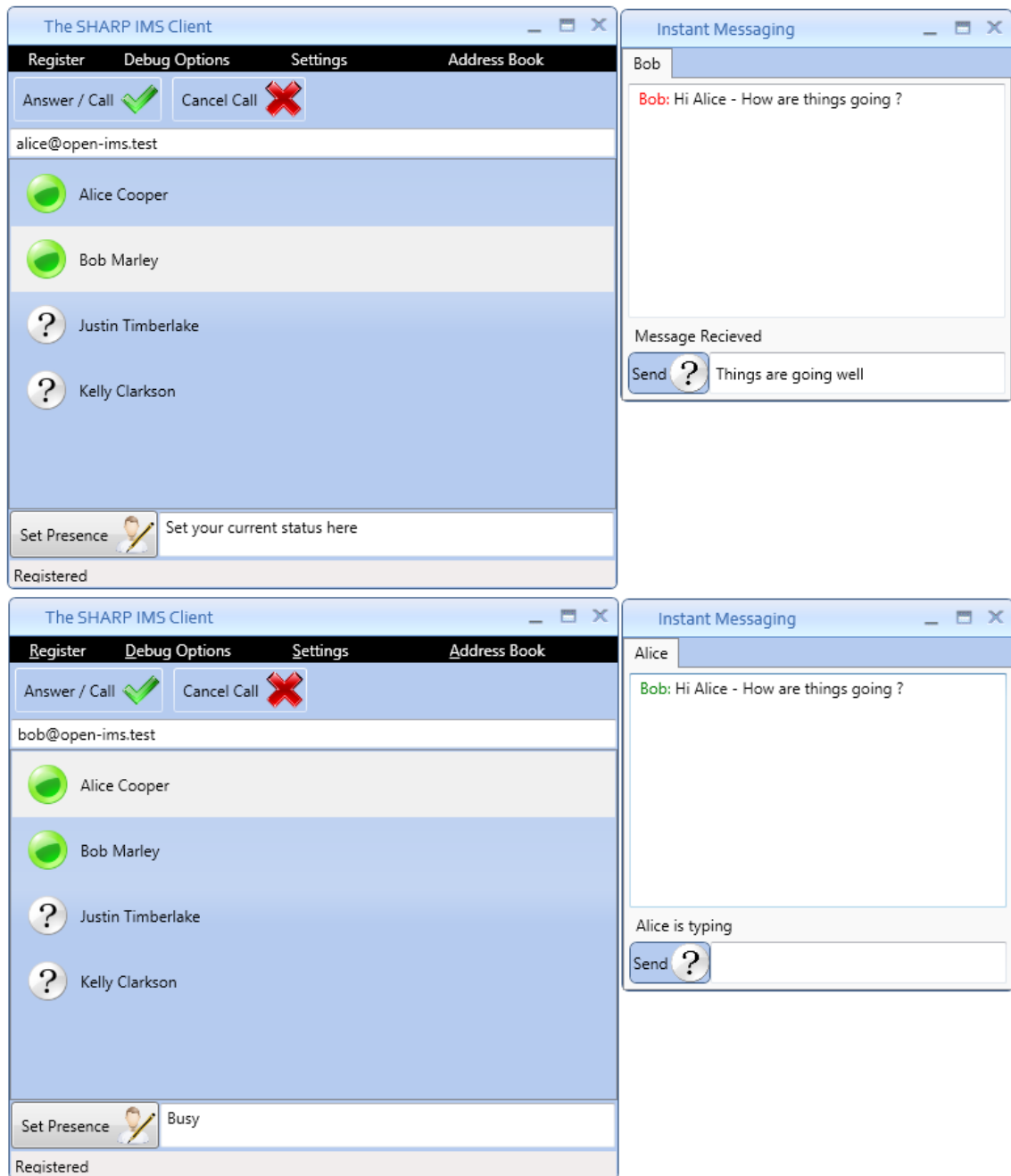


Figure 6.6: IMS Client messaging functionality.

In addition, the following functionality was optional but recommended to enhance testing and research:

- Full logging of SIP traffic
- Ability to programmatically time and call functions in an automatic fashion
- Support a flexible “options” or “preferences” framework that can easily be extended
- Support a network based contact list

During development of the client the above features were added to aid development. They were very useful during the implementation stage where new features of the client were tested, as well as during the testing phase where the overall framework’s functionality was being examined. The network contact functionality was useful for storing a single set of contacts in the network. Each client instance could access these contacts allowing multiple presence updates to be viewed across all clients without having to reconfigure each client instance individually. The client code used in this thesis has been released under an open source library for other researchers to use and extend [106].

6.4.2 Web browser

In order to remain compatible with a wide range of browsers standard libraries such as D3 were used for client side of the web interface. This allowed powerful, flexible interaction with the USPS [117]. These libraries were used for the front-end of the USPS and mainly involved the display of information. Most of the logic and service functionality was deployed in the back-end, i.e. on the server running the USPS.

This allowed a range of browsers to be used for this portion of the testing framework. The web browser was used to allow the end user to choose and edit their preferences with regards to their desired service execution scenarios. Superficial testing using various browsers (Firefox, Opera, Chrome) was carried out but has been omitted as the tests were successful and have no further bearing on the functionality or performance of the proposed architecture.

Chrome, a web browser developed by Google, was chosen to be used as the standard web client and all end user interactions with the USPS were done through this browser [130].

6.5 Measuring and utility software

While the previous sections have described the main software used for providing the needed services in the test-bed, several different suites of software were used for measuring or capturing the results from the test-bed. These are highlighted below.

6.5.1 Smokeping

Smokeping is a program that repeatedly sends Internet Control Message Protocol (ICMP) packets from the monitoring host to the monitored targets [122]. It keeps track of the round trip time and uses it to keep track of the latency between the hosts. The results are stored in a round robin database (RRD) and charts can be drawn from this information. This was used to provide a baseline measurement of latency between the different hosts in the test bed.

6.5.2 Wireshark

Wireshark is a popular, open-source packet capturing tool. It can capture and dissect many different network protocols. It was used to capture and display the SIP traffic between all components of the test-bed. It has built in protocol handlers which can automatically pull out and show the various fields in a network packet that correspond to known values. An example of this can be seen in Figure 6.7 where a SIP packet is being examined.

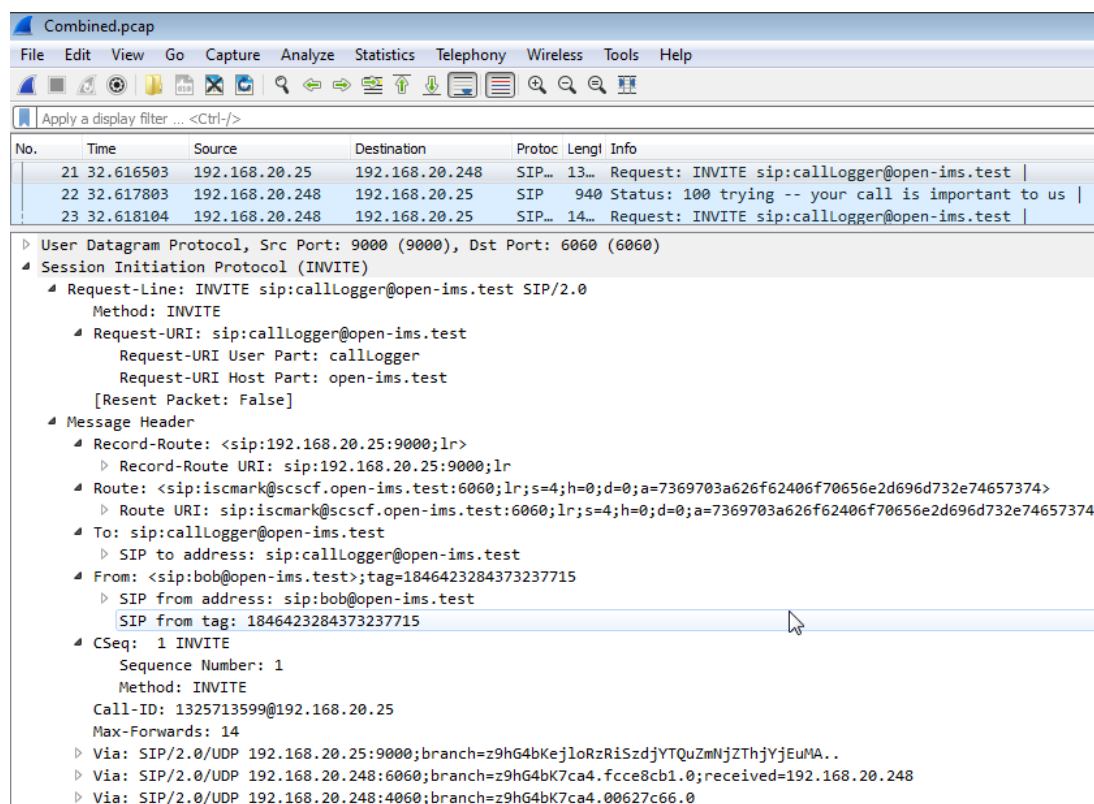


Figure 6.7: Wireshark GUI with a SIP packet capture.

6.5.3 Call-flow Sequence Diagram Generator

The “Call-flow Sequence Diagram Generator” is an open source program to generate signalling diagrams from captured packet dumps [131].

It is comprised of several different shell and awk scripts that take a packet capture file and produce a sequence diagram. The packet capture files produced by Wireshark are compatible with this program, and it is very useful to debug SIP call flows.

While the majority of the program functioned as expected, small modifications were needed to allow it to run on more recent versions of Linux (as it was last updated in 2014).

6.5.4 IMS Bench / SIPp

SIPp is an open source traffic generator for the SIP protocol [132]. It has been extended to form IMS Bench, by adding the necessary SIP extensions to interact with the IMS. Further modifications were made to allow scenarios that include a large number of users. It includes tools to monitor and report on the resources of the individual VMs taking part in the testing, and is capable of generating a final report that complies with the ETSI TS 186 008 [128].

It is configured through an XML file which details the parameters of the overall test as well as parameters for the individual scenarios under test. An extract from this XML configuration file can be seen in Figure 6.8 on the following page.

While the IMS client discussed previously was used for the majority of development and testing, the UE functionality included with IMS Bench was utilised for the performance tests that required automatic, scripted interactions that could be repeated frequently in a consistent manner.

```

<?xml version="1.0" encoding="ISO-8859-1" ?><configuration>
  <!-- Test System Parameters -->
  <!-- Registration Only -->
  <param name="number_test_systems" value="1"/>
  <param name="prep_offset" value="2000"/>
  <param name="rand_seed" value="0"/>
  <param name="report" value="1"/>
  <param name="log" value="1"/>
  <param name="transient_time" value="30"/>
  <param name="max_time_offset" value="250"/>

  <!-- Scenario Parameters -->
  <scen_param name="RingTimeDistr" value="exponential"/>
  <scen_param name="HoldTime" value="120000"/>
  <scen_param name="RingTime" value="5000"/>
  <scen_param name="RegistrationExpire" value="1000000"/>
  <scen_param name="PMMDDataSize" value="140"/>
  <scen_param name="PMMDDataSizeDistr" value="uniform"/>
  <scen_param name="HoldTimeDistr" value="exponential"/>
  <!-- Scenario -->
  <scenario name="ims_rereg" max_ihs="0.1"/>
  <scenario name="ims_dereg" max_ihs="0.1"/>
  <scenario name="ims_msgc" max_ihs="0.1"/>
  <scenario name="ims_msgs"/>
  <scenario name="ims_reg" max_ihs="0.1"/>
  <scenario name="ims_uac" max_ihs="0.1"/>
  <scenario name="ims_uas"/>
</configuration>

<!-- Pre-Registration Phase -->
<run cps="10" max_calls="3200" distribution="constant" sync_mode="off"
1000">
  <scenario name="ims_reg" ratio="100"/>
</run>

```

Figure 6.8: Extract from IMS Bench XML configuration file.

6.6 Discussion

This chapter has detailed the necessary components required to set up a test-bed to investigate the proposed architecture. The components used provided a fully functioning environment for demonstrating the prototype's feasibility and a suitable platform to investigate whether or not the overall architecture is viable. The following chapters contain details on the functional and performance tests performed using this test-bed, as well as the resulting data gathered from them.

Chapter 7

Test Results

The test results in this thesis have been divided into two main sections. The first section examines the results from testing the functionality of the proposed architecture. It looks at the signalling involved between the components and ensures that the correct results are achieved, i.e. the architecture functions as expected.

The second section details performance testing, and ensures that the functionality demonstrated in the first section is realistically achievable and does not hinder the scalability of the architecture. Together, these test results indicate the overall feasibility of the project. All signal diagrams shown in this chapter are generated from actual packet flows captured from the test-bed.

7.1 Functional Testing

The test-bed was tested for correct operation with and without the proposed enhancements. This set a baseline and ensured that similar results were obtained under both architectures. To allow for repeatable results with a minimum of configuration changes between architectures, another Service Profile (SP), called “*enhanced_services*”, was created via the HSS web GUI.

This SP contained the necessary iFC to allow the new architecture to function (i.e. it redirected requests to the SCIM). To enable or disable the new functionality the new elements were started or stopped and the desired SP was selected for the specific user account depending on the test scenario. This limited the amount of changes required between test runs and ensured that the changes were done in a repeatable fashion.

7.1.1 Original Functionality

The first scenario tested was the developed IMS client registering with the IMS network. The registration phase takes part after IP connectivity has been established and the local address of the P-CSCF has been discovered.

The IMS client sends a SIP REGISTER request to the P-CSCF, and receives a 401 UNAUTHORISED response. The client then calculates the correct response for the challenge contained in the 401 response and resubmits the SIP REGISTER request including this calculated response.

A 200 OK response indicates that the client has successfully registered on the network, and can begin utilising services. The signal diagram for this can be seen in Figure 7.1.

The next set of functionality tested was the presence subsystem. This involved the client publishing its own presence information as well as subscribing to another end user's status information. Once the subscription had been established, notifications of any updates were forwarded to the client whenever another client or contact made changes to their presence information.

These changes were made by utilising the SIP PUBLISH request which is redirected to the Presence Server (PS). The PS maintains the current state of the end user. The signal diagram showing the path followed for a simple SIP PUBLISH can be seen in Figure 7.2.

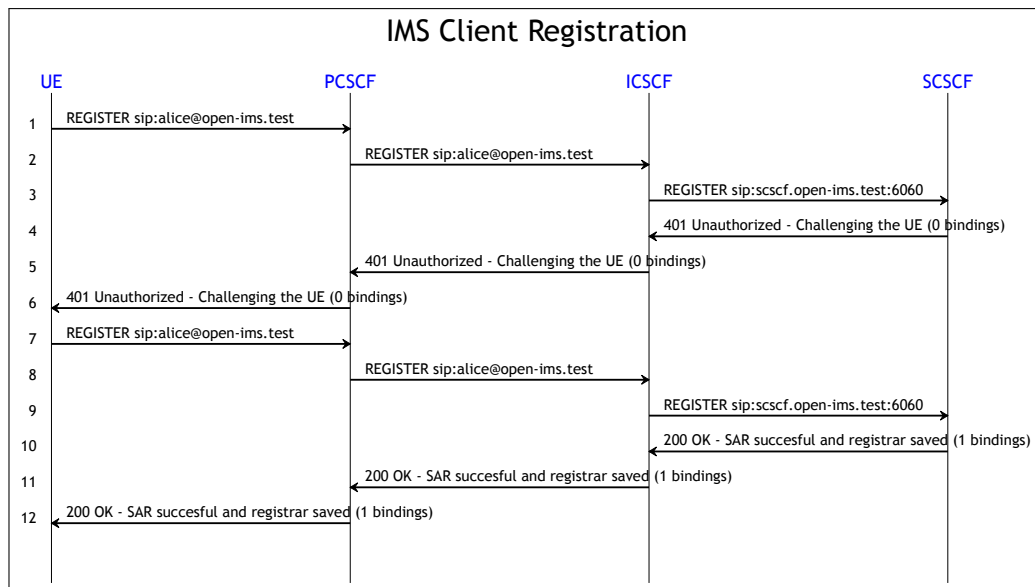


Figure 7.1: Signal diagram showing client IMS registration.

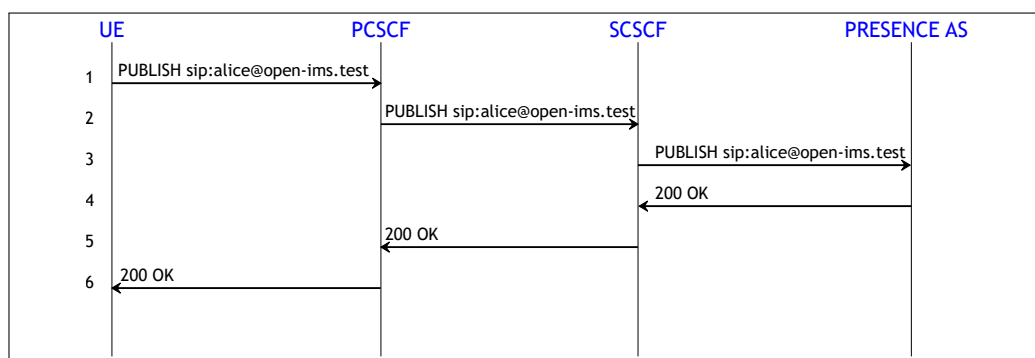


Figure 7.2: Signal diagram showing client IMS presence PUBLISH request.

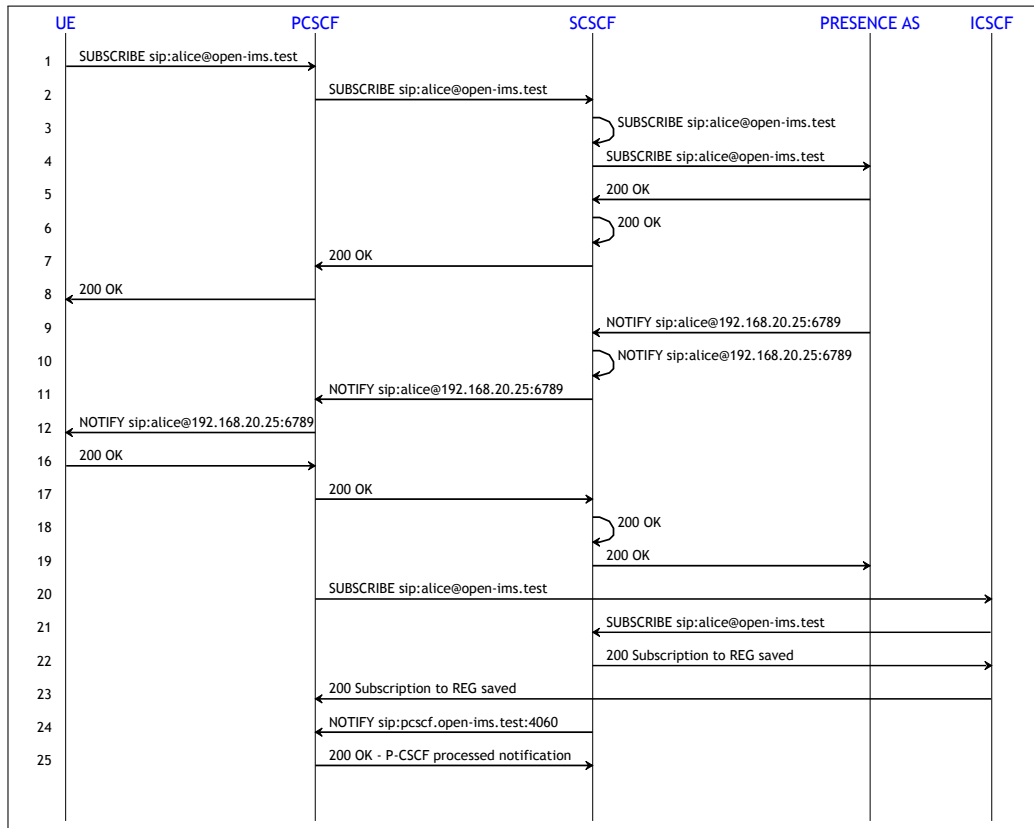


Figure 7.3: Signal diagram showing client IMS presence SUBSCRIBE request.

Figure 7.3 is more complicated, and shows the UE both subscribing to and receiving presence information, as well as the special case of the P-CSCF subscribing to the registration state event package.

This allows the P-CSCF to be informed by the S-CSCF on any changes in the end user’s registration, and allows the P-CSCF to be notified in real time for any changes to the list of registered Public User Identities (PUIs). For this particular subscription request the Event header field contains the value “*reg*” and the Accept header field is populated with the value “APPLICATION/REGINFO+XML”.

A similar message flow is used to ensure that the CS is kept up to date with any status changes in the users it is configured to track.

The next set of tests verified the communication capabilities of the client. Both video and audio calls were established between two instances of the developed IMS client, and instant messages using the SIP MESSAGE request were exchanged.

The signal diagrams for a voice call can be seen in Figure 7.4, and an example of a SIP request with the SDP for a video call can be seen in Figure 7.5.

Several different stub IMS services were developed as part of the framework. These stub services were tested for correct functionality both with and without the proposed architecture. For example, a call-logging service which records the metadata about each SIP session before forwarding the SIP INVITE request to its destination was one of the services developed.

An example signal diagram for a SIP INVITE request that has been forwarded to the call logger server by the normal iFC mechanism can be seen in Figure 7.6. This demonstrates the normal service chain that is possible with the original IMS framework, and can be compared to the signal flow in Figure 7.12 later on.

The successful execution of the above operations showed that the developed client and test bed were all operating correctly.

7.1.2 Enhanced Functionality

Once it was established that the developed client and the test-bed with the standard IMS architecture was functioning in the correct fashion, testing of the proposed architecture could begin.

The proposed architecture was configured as described in the previous section, moving from the standardised architecture to the enhanced architecture with minimal changes. The main configuration change implemented when moving between the standard and proposed architecture was the configuring of the iFC to redirect service requests to the SCIM.

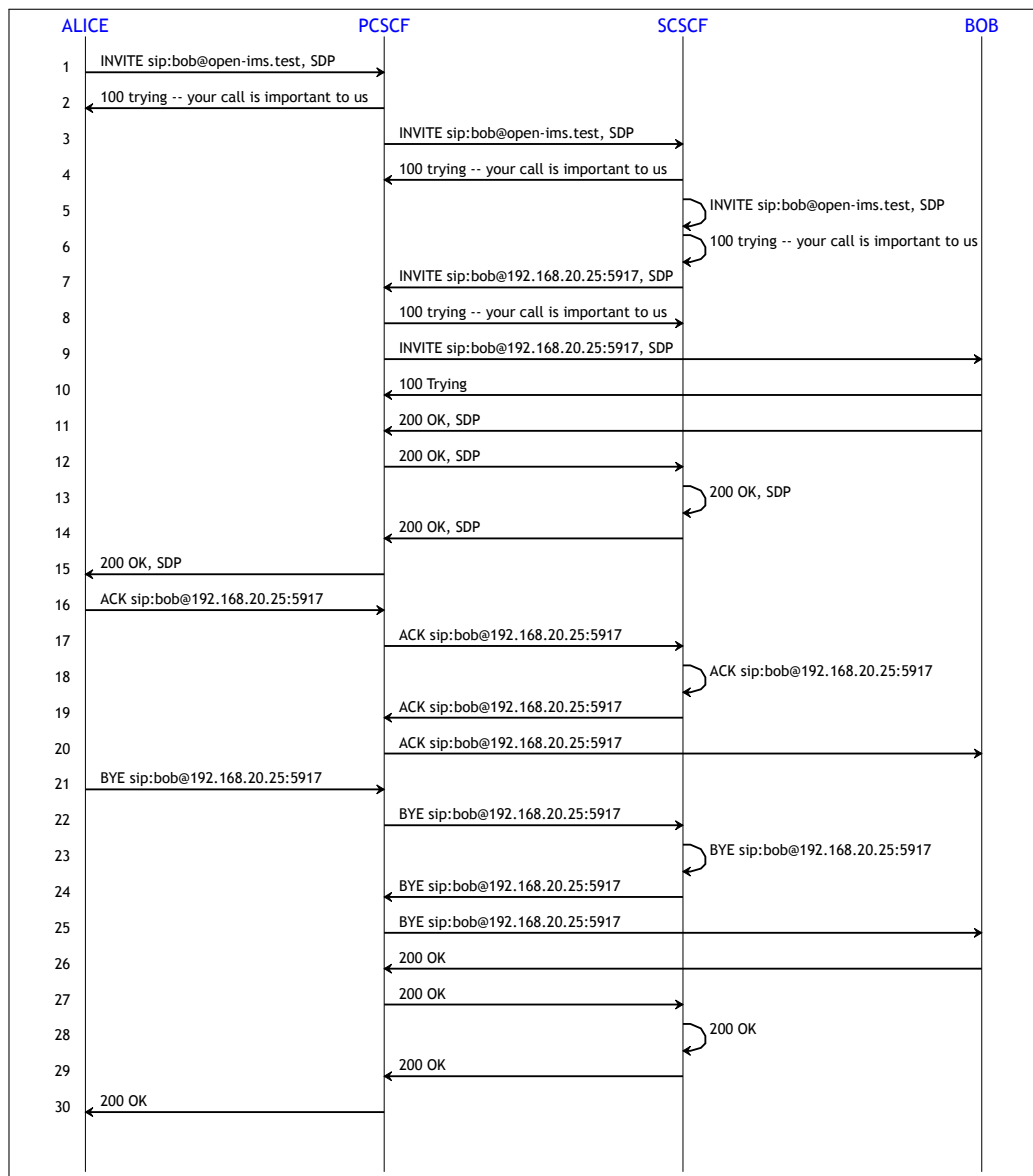


Figure 7.4: Signal diagram showing voice call between two IMS clients.

```
INVITE sip:bob@open-ims.test SIP/2.0
To: <sip:bob@open-ims.test>
From: <sip:alice@open-ims.test>;tag=194946565309933898
CSeq: 1 INVITE
Call-ID: 171549612@192.168.20.25
Max-Forwards: 70
Via: SIP/2.0/UDP 192.168.20.25:6789;branch=z9hG4bKPHNpcDpib2JAb
Contact: <sip:alice@192.168.20.25:6789>
Content-Type: application/sdp
User-Agent: IMS-CLIENT
Route: <sip:orig@scscf.open-ims.test:6060;lr>
Content-Length: 308

v=0
o=- 0 0 IN IP4 192.168.20.25
s=IMS Call
c=IN IP4 192.168.20.25
t=0 0
m=audio 30666 RTP/AVP 3 0 101
b=AS:64
a=fmtp:101 0-11
a=rtpmap:3 GSM/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
m=video 21629 RTP/AVP 96
b=AS:128
a=fmtp:96 profile-level-id=0
a=rtpmap:96 H263-1998 /
```

Figure 7.5: SIP INVITE request with SDP for video call.

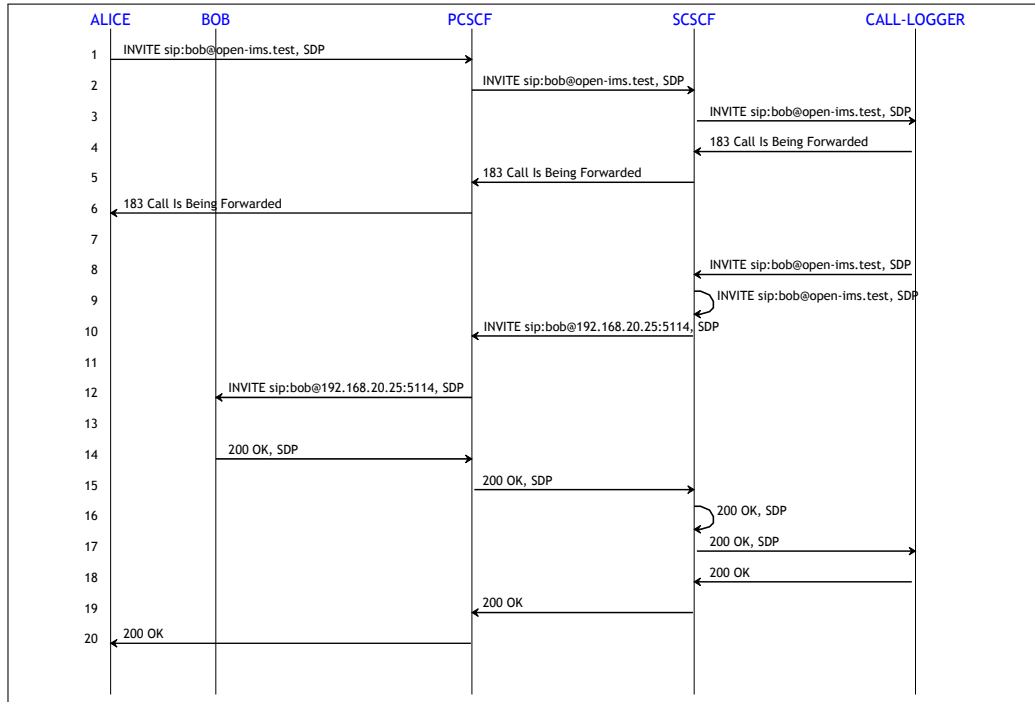


Figure 7.6: SIP INVITE redirected to call metadata log service.

The configuring of the iFC was done through the web front-end of the HSS. The network operator may choose to include only certain messages for enhanced handling by the SCIM. For example, a Trigger Point (TP) that only routes SIP INVITE or SIP MESSAGE requests can be seen in Figure 7.7.

A particular service profile was created with the necessary iFC for the standard architecture and one for the enhanced architecture to redirect requests to the SCIM. In order to identify correct contextual service triggering, three different test cases or scenarios were proposed.

The first scenario tested the ability of the framework to redirect based on the To and FROM fields of a SIP message. The core of this functionality can be duplicated in the original architecture through the existing iFC mechanism, and it was chosen to serve as the first starting point in this evaluation to ensure correct operation of all components.

Not	☒	SIP Method	PUBLISH	Delete
OR				
Request-URI				
AND				
Not	☒	SIP Method	SUBSCRIBE	Delete
OR				
Request-URI				
AND				
Not	☒	SIP Method	REGISTER	Reg ☐ ReReg ☐ DeReg ☐ Delete
OR				
Request-URI				
AND				
Not	☐	SIP Method	INVITE	Delete
OR				
Not	☐	SIP Method	MESSAGE	Delete
OR				
Request-URI				

Figure 7.7: Trigger Point redirecting certain requests to SCIM.

The second scenario tested was the ability of the framework to trigger different services from the same SIP request based on the current time of day. This functionality cannot be duplicated with the current standardised architecture without the use of a third party service.

The final scenario tested was a scenario identified in the 3GPP study as a scenario that is not possible under the current standardised architecture. This scenario involves several different services and a specific order of service triggering. These scenarios were chosen to determine if the architecture was feasible for its purpose, i.e. whether or not it is possible to have dynamic service routing controlled by the end user's preferences that takes into account the current specific context of the user.

Contact differentiation

The first test case was possible under the original architecture. As such an iFC was created to statically redirect all requests from *alice@open-ims.test* that were originally directed towards *bob@open-ims.test*. This iFC redirected these requests to *voicemail@open-ims.test*, which is a Public Service Identity (PSI) for the voicemail service. The signal flow for this scenario can be seen in Figure 7.8.

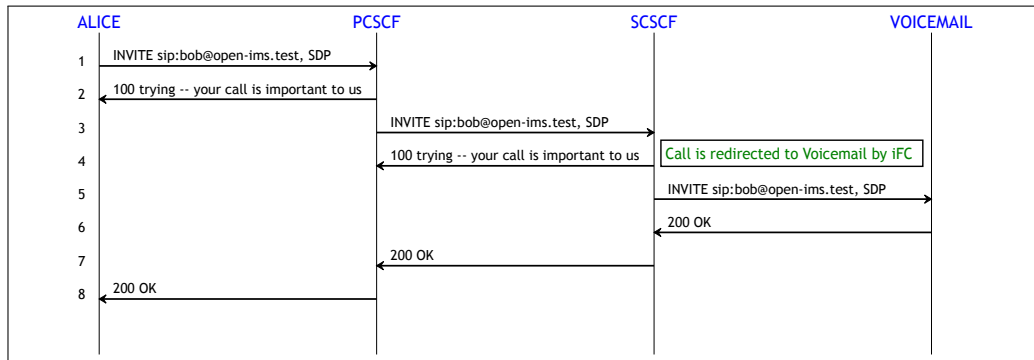


Figure 7.8: Static redirect to voicemail through contact differentiation iFC.

This test case was duplicated by using the new mechanisms provided in the proposed architecture. Namely, a service chain specifying the required action was created using the web interface of the USPS under Alice's profile.

This was configured to redirect all calls from Bob to the voicemail service. This was a fairly simple example so each step in the user interface to create this service flow can be shown. These steps will be omitted for the more complex service flows used in the other test cases. The steps taken to create this service flow can be seen in Figure 7.9 and the signal diagram for this scenario can be seen in Figure 7.10.

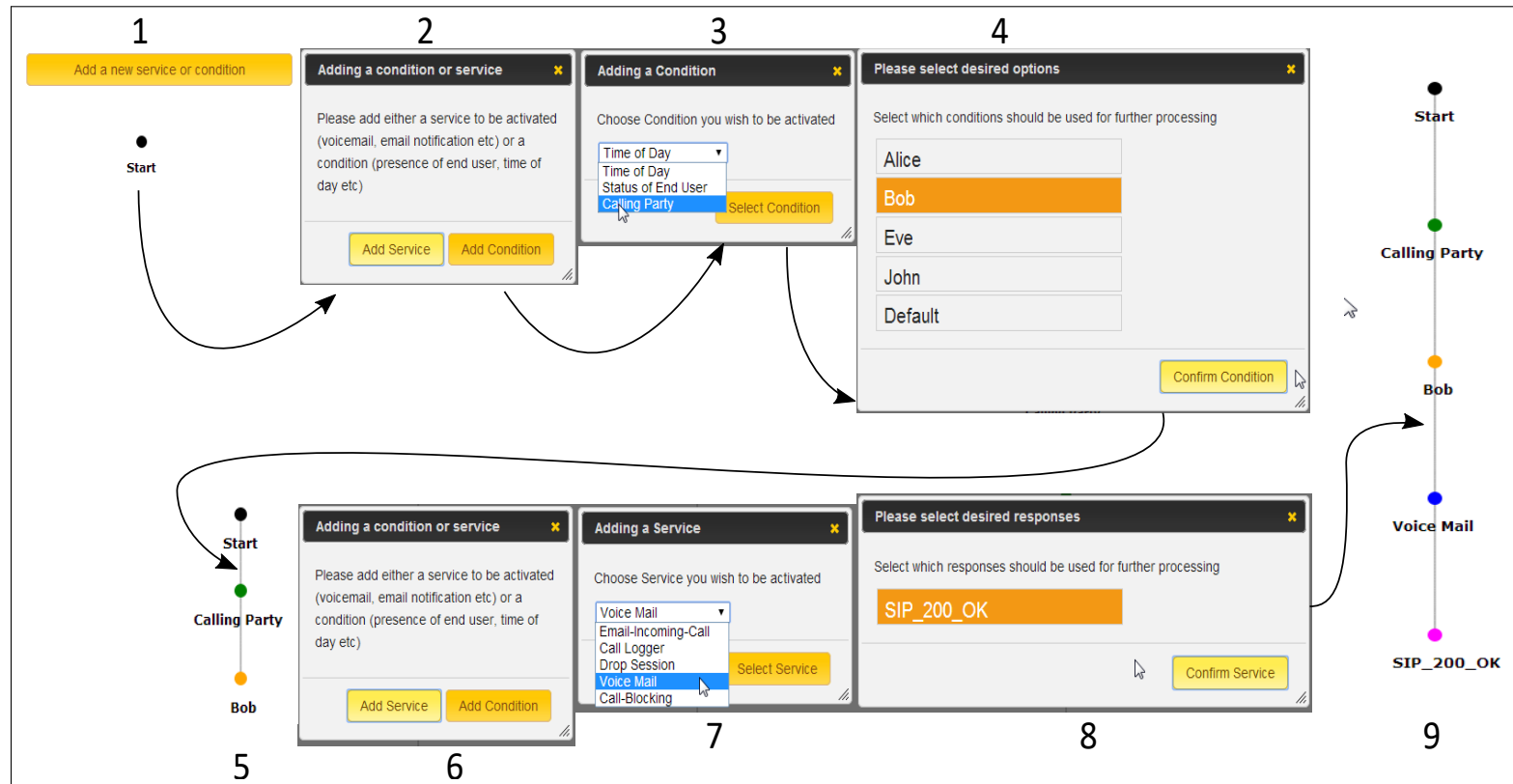


Figure 7.9: Creating a contact specific redirection to voicemail through the USPS.

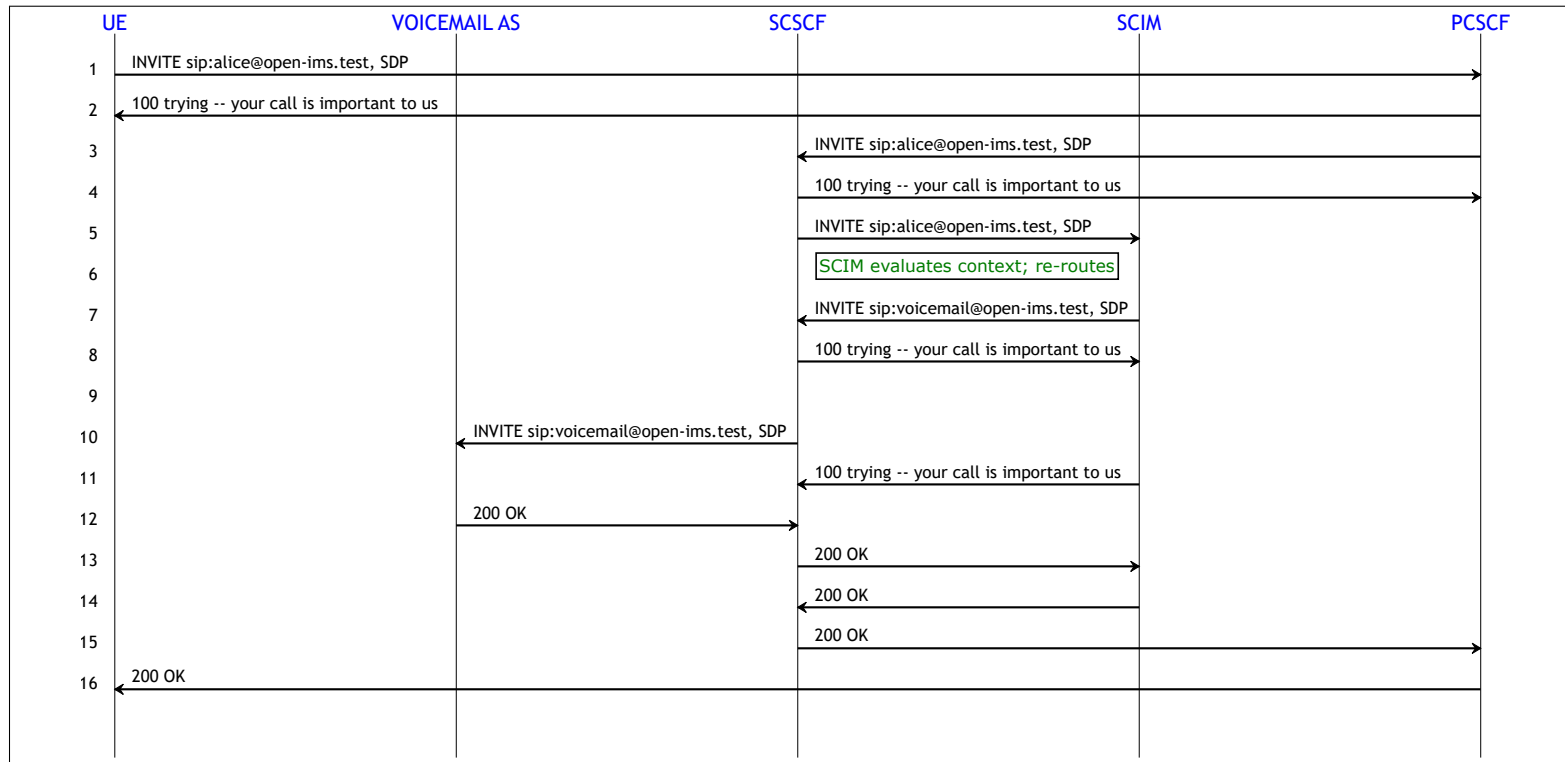


Figure 7.10: Redirect to voicemail through contact differentiation via SCIM.

Time differentiation

The next test case was chosen to closely resemble the first test case, but with the added condition that the context for making the routing decision had to come from outside the SIP message, i.e. the routing decision was taken based upon external information.

This is not possible in the current standardised IMS architecture, although it is possible by utilising a custom third party AS. The web interface of the USPS was once again used to setup this test case. The following service chain was defined:

- If the current time falls into business hours (between 08:00 and 17:00), allow Alice to receive the call as normal.
- If the current time falls outside business hours (before 08:00 and after 17:00), and the caller is Bob, redirect to the voicemail service.

The scenario setup can be seen in Figure 7.11. The signalling diagram is identical to the previous signalling diagram (Figure 7.10) and as such was omitted. The key difference was that the decision taken was no longer based on the caller's ID but rather the current time. The new architecture provides more functionality than the standard architecture in achieving this test scenario.

3GPP specified scenario

The last test case chosen was a specific scenario that the 3GPP chose in their study [46]. Here they note that it is currently not possible to achieve this scenario with the standardised architecture, and as such it is an ideal test case to demonstrate the additional functionality.

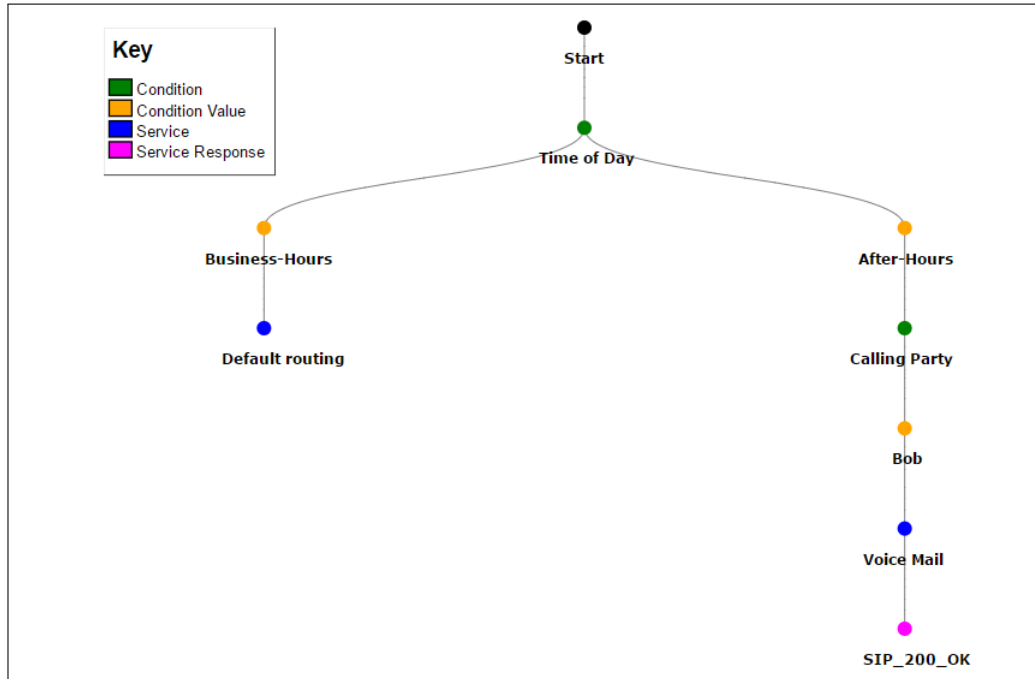


Figure 7.11: Time based service routing.

The proposed scenario, taken from the study, is as follows [46]:

- If the end user's presence is "*do not disturb*", the IMS will bar all the incoming requests, but record the information of requester (i.e. requester's Caller ID) and send an e-mail notification to alert the user.
- If the end user's presence is not "*do not disturb*", all incoming requests will be screened against a black list. If the requester is in the black list, the incoming request will be barred while the information of requester (i.e. requester's Caller ID) is logged and an e-mail notification is sent to alert the user.
- If the end user's presence is not "*do not disturb*" and the incoming requester is not in the black list mentioned above, route the SIP request normally.

This scenario can now be executed successfully with the proposed architecture in place. A call diagram showing the signalling flow for the first part of

the scenario is presented in Figure 7.12. Several acknowledge messages (i.e. SIP ACK and 100 TRYING responses) as well as certain requests have been omitted for clarity.

7.1.3 Discussion

The tests in this section were undertaken to evaluate the proposed architecture on a functional level. No functionality has been lost compared to the original architecture, and the tests show that the proposed architecture adds additional functionality.

The architecture successfully provides a framework to enable dynamic contextual routing, adding new functionality and increasing the flexibility of the standardised architecture. The tests were successfully conducted using a prototype implementation in a test-bed based on the layout of a typical IMS core network. This shows that the architecture is feasible and can be deployed along side the existing IMS architecture without requiring large scale changes.

While the desired functionality has been achieved on a functional level, it still needs to be verified that it has not greatly overloaded any component or added an unacceptable amount of delay.

These issues are examined in the next section which details the impact that the proposed architecture has on the session establishment delay to evaluate whether or not it is within suitable bounds.

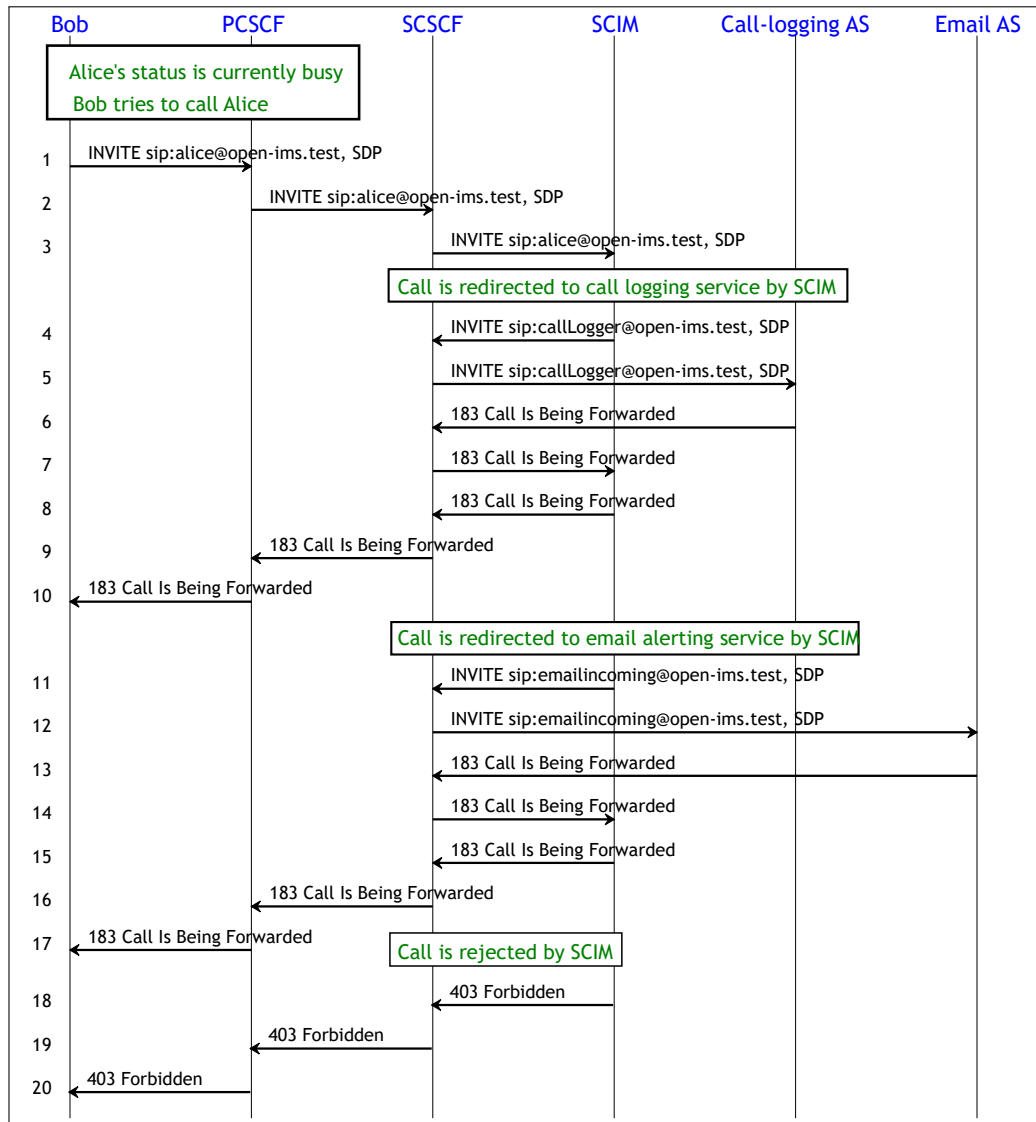


Figure 7.12: SIP Signal Flow for 3GPP scenario.

7.2 Performance Testing

Section 7.1 has discussed the functional tests that verified that the architecture was operating as expected. Once the functionality was verified, the performance of the proposed system was examined. As discussed previously, the intention here was not to measure the overall capacity of the proof-of-concept implementation, but rather to ensure that the additional functionality did not increase the end to end session setup delay past acceptable bounds.

The ETSI have developed specifications for testing how IMS systems behave under load [128], and while we were not interested in the overall capacity of the demo implementation, they served as useful guidelines for performing the below tests. These specifications include sections on how to apply load to the System under Test (SuT) in changing steps, as well as requiring settling time to allow components to stabilise at the start of each run within the test. While the specification is fairly involved, it has been followed where applicable in both methodology and nomenclature.

Each virtual server running a component of the test-bed was considered a System under Test (SUT), and there was a separate virtual server acting as the Test System (TS). The TS was responsible for simulating user endpoints and generating the SIP requests towards the IMS core network. These requests form part of a scenario, and the overall Scenarios Attempted Per Second (SAPS) is a metric of interest when performing these tests. Scenarios can involve a number of different sessions or message types, hence the term *scenario* is used instead of the term *call*.

The specification also indicates that there should be a dedicated setup period during which the users are registered with the IMS as well as a stir phase to randomise the starting conditions of the test. During the stir phase a low number of scenarios are executed for a pre-defined period, and these normally include registration, de-registration and re-registration scenarios. After these two periods, the load is applied to the SUT in defined steps. Each step is executed for a defined time period, after which the next step in

load is applied. As the test reports are fairly verbose (in the order of 20 pages each) they have been excluded. Only the key findings are discussed in this chapter. An example of these test reports has been included as Appendix A.

7.2.1 Registration

Normally the registration of the end users is handled during the preamble or setup stage of the tests, and is mostly discounted as often the interest lies in the number of messages or calls that a system can handle. However, we were interested in ensuring that there was no deviation in performance of the the actual registration operations when comparing the proposed and standardised architectures. A registration preamble was conducted with a few thousand registrations (4000) to provide a base for the de-register and re-register scenarios. The scenarios executed during the benchmark runs consisted of either registering a user, de-registering a user, or re-registering a user.

Configuration

A target of 25 calls per second (CPS) was set for the registration scenario. The intention here was to exceed the capacity of the test-bed near the end of the test. Experience with the test-bed showed that a value of 20 CPS can be supported, and this is slightly higher than the reported value of 15 CPS in literature [133].

The value of 20 CPS for registration for the HSS has been reported as a limitation by some users of the open source Java based version, and a more performant version (CHeSS) is provided under a paid licence [134]. This value is also affected by the resources available to the MySQL server and the HSS. For our purposes the free open source version was deemed sufficient.

We did not need to push the test bed further as we were more interested in

a comparative result rather than the overall capacity of the SuT or the total number of operations successfully completed.

The IMS Bench program was configured to trigger 10 CPS of the registration scenario (`ims_reg`) with a constant distribution until the maximum number of scenarios executed reached 4000. The stir phase of the test is intended to slightly randomise the starting conditions of the test, and ensure that certain functions have been exercised at least once to improve accuracy due to caches being pre-warmed etc. The stir step was a mix of registration, de-registration and re-registration scenarios executed at 5 CPS. The duration of the step was set to 5 minutes. The call distribution was configured to a Poisson distribution as this is widely used to model call arrival rates in telecommunication literature. The percentage mix of the scenarios was as follows: Registration 40 %, De-registration 40 %, Re-registration 20 %. The benchmark phase consisted of the same ratio of scenarios as the stir phase, but the number of steps were increased. The duration of each step was kept at 5 minutes, with an increase of 5 CPS per step. The targeted number of CPS was 25. This gave 5 distinct steps. This resulted in a test run that took roughly 36 minutes and 50 seconds.

Results

This test was repeated with or without the proposed architecture in place. For this scenario, the difference between the standard and proposed architecture was the addition of a iFC that did not affect the routing of the SIP register message.

The intention here was to determine if this additional iFC played a significant role in request delay. Thus we needed a scenario where only the iFC was triggered, and the requests were not forwarded to the SCIM.

A feasible scenario for the network operator would be excluding the SCIM from the registration procedure. This is one of the more likely scenarios

Average CPU Usage (%)	Step 1	Step 2	Step 3	Step 4	Step 5
P-CSCF	0.09	0.11	0.15	0.15	0.20
I-CSCF	0.15	0.24	0.30	0.39	0.44
S-CSCF	0.14	0.22	0.29	0.41	0.51
HSS	1.88	2.64	4.67	5.94	8.53
Test-Manager	0.43	0.39	0.44	0.46	0.79

Table 7.1: CPU usage - Registration over standard architecture.

Average CPU Usage (%)	Step 1	Step 2	Step 3	Step 4	Step 5
P-CSCF	0.11	0.13	0.16	0.2	0.22
I-CSCF	0.16	0.24	0.31	0.36	0.43
S-CSCF	0.16	0.26	0.34	0.41	0.55
HSS	2.18	2.67	4.82	6.28	9.8
Test-Manager	0.33	0.28	0.32	0.31	0.16

Table 7.2: CPU usage - Registration over proposed architecture.

where the SCIM is excluded, as it is envisaged that the additional functionality is more valuable for other types of messages such as SIP INVITE or SIP MESSAGE requests. Here it serves as a model for the network operator choosing to only route certain messages to the SCIM.

As can be seen in Table 7.1 and Table 7.2 the CPU of the virtual servers had plenty of spare capacity, and there was no significant difference between the two scenarios.

The HSS showed the highest amount of CPU usage as expected due to its role in the registration process. As even the HSS had plenty of spare CPU capacity it indicates that the CPU was not a bottleneck in either scenario, and played no significant role in the evaluation of the additional iFC.

Similarly, the free memory available on each virtual server can be seen in Table 7.3 and Table 7.4. There was more than enough free memory available for all the virtual servers involved in the test and it was not significantly different between the two scenarios.

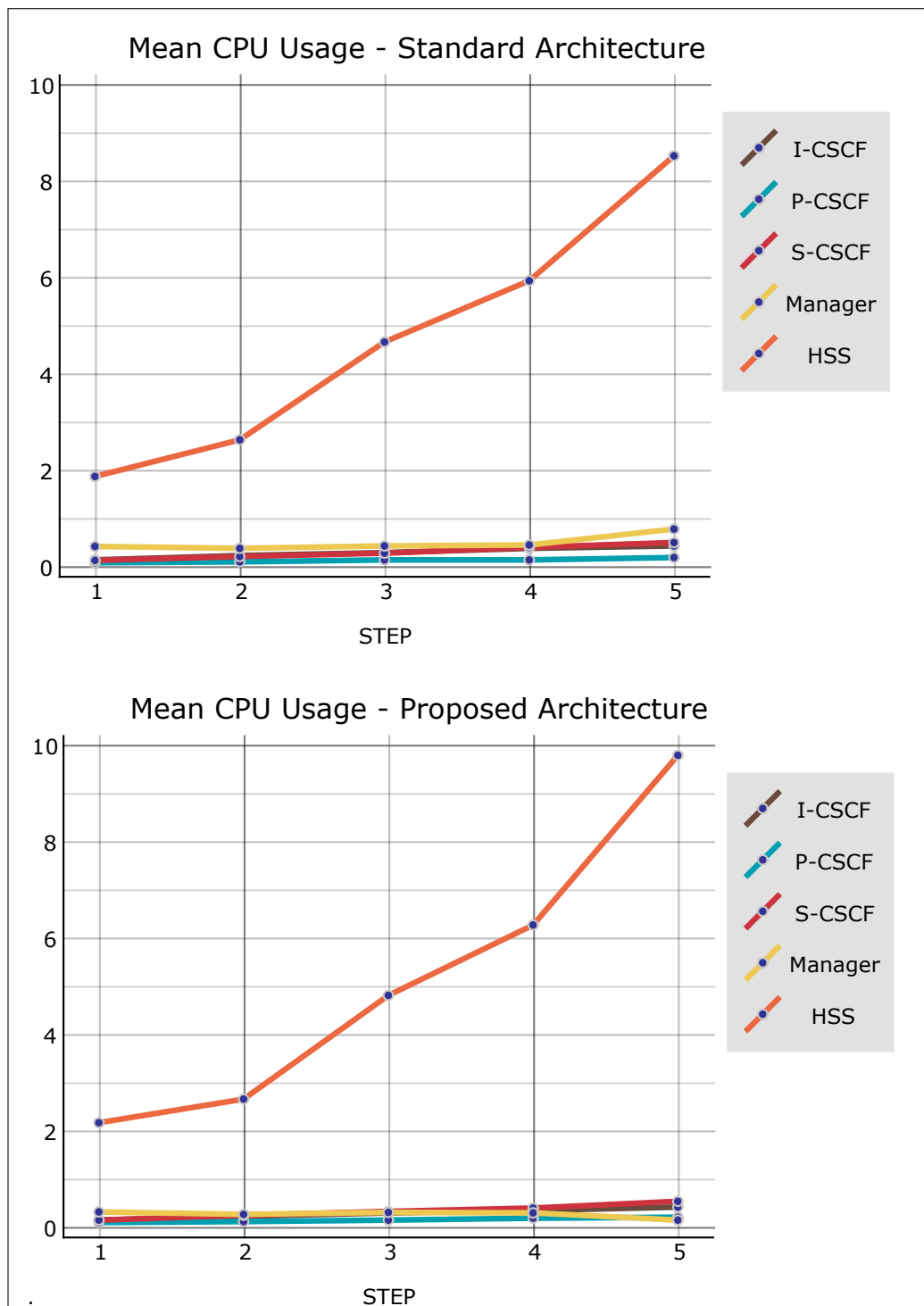


Figure 7.13: Average CPU usage during registration.

Average Free Memory (MB)	Step 1	Step 2	Step 3	Step 4	Step 5
P-CSCF	3765.88	3765.63	3764.56	3763.20	3761.35
I-CSCF	3769.21	3768.74	3767.69	3766.33	3764.94
S-CSCF	3728.63	3723.79	3716.50	3707.82	3695.96
HSS	1997.42	1976.03	1879.67	1684.38	1534.77
Test-Manager	528.54	527.48	526.41	525.30	523.87

Table 7.3: Free Memory - Registration over standard architecture.

Average Free Memory (MB)	Step 1	Step 2	Step 3	Step 4	Step 5
P-CSCF	3766.03	3765.62	3764.59	3763.22	3761.64
I-CSCF	3773.09	3772.56	3771.61	3770.27	3768.62
S-CSCF	3723	3716.98	3708.69	3697.77	3681.89
HSS	1958.05	1954.6	1878.62	1728.99	1591.92
Test-Manager	551.23	550.23	549.24	547.97	546.49

Table 7.4: Free Memory - Registration over proposed architecture.

Again the HSS is the virtual server with the most impact. There is a large difference between the amount of free memory of the IMS components and the test-manager, as the test-manager was provisioned with a much smaller instance type with a lower total capacity of memory. As can be seen in Figure 7.14, as the load increases the amount of free or available memory on the HSS decreases, indicating more memory is consumed as more users are registered per second.

The resource usage results indicate that there were no hardware capacity bottlenecks during the test execution, and the results should not have been impacted by a limit in either processing power or available memory.

While this may appear an inefficient use of the server resources and we could have used a lower capacity virtual server instance type, we deliberately chose a higher capacity instance type to avoid artificially limiting any performance. As the CPU and memory usage is similar for the remainder of the tests they have been omitted unless they are note worthy.

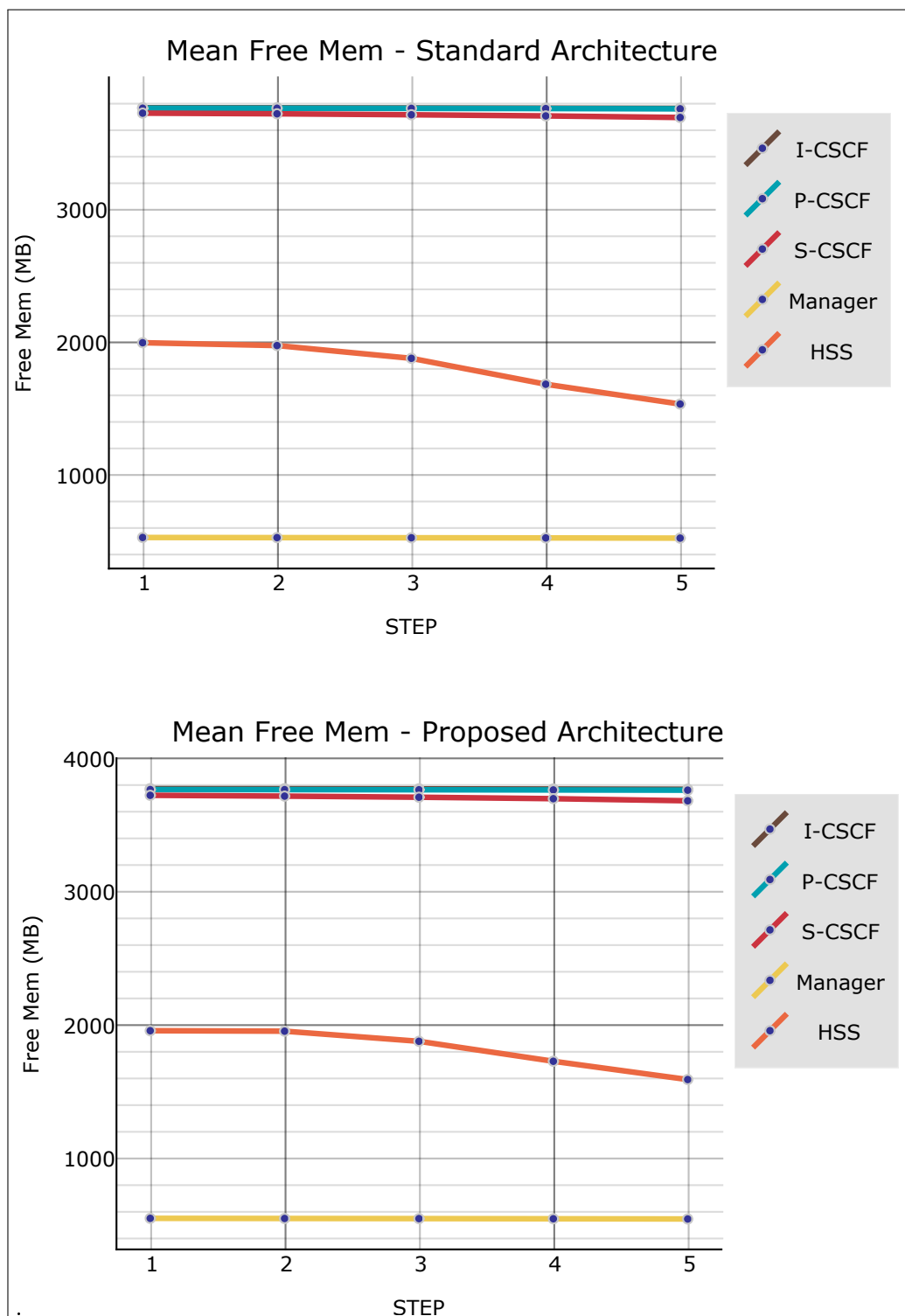


Figure 7.14: Average Free Mem usage during registration.

Step	Requested Load (CPS)	Mean (ms)	Minimum (ms)	90th Percentile
1	5	11.87	5.89	11.1
2	10	10.40	5.52	11.0
3	15	10.74	5.57	11.0
4	20	27.84	5.43	11.3
5	25	27.41	5.47	12.1

Table 7.5: Delay between REGISTER and 401 over standard architecture.

Step	Requested Load (CPS)	Mean (ms)	Minimum (ms)	90th Percentile
1	5	15.04	5.75	11.0
2	10	10.50	5.52	10.9
3	15	15.75	5.45	10.9
4	20	21.07	5.48	11.4
5	25	29.93	5.34	12.8

Table 7.6: Delay between REGISTER and 401 over proposed architecture.

In addition to the CPU and memory metrics the delay between the first SIP REGISTER request and the SIP 401 UNAUTHORISED response for all scenarios was captured, and can be seen in Table 7.5 and Table 7.6.

Discussion

The proposed architecture does not significantly change the registration process, and as such the performance of user registration between the two architectures was expected to be similar.

The additional iFC needed for the new architecture plays a negligible role in determining the overall registration time, and does not noticeably affect system resources or performance. This shows that it is possible to add the SCIM to the existing network architecture by utilising the existing iFC mechanism without negatively impacting the original behaviour of the network. Services that are not routed through the SCIM will continue to operate as before without penalty.

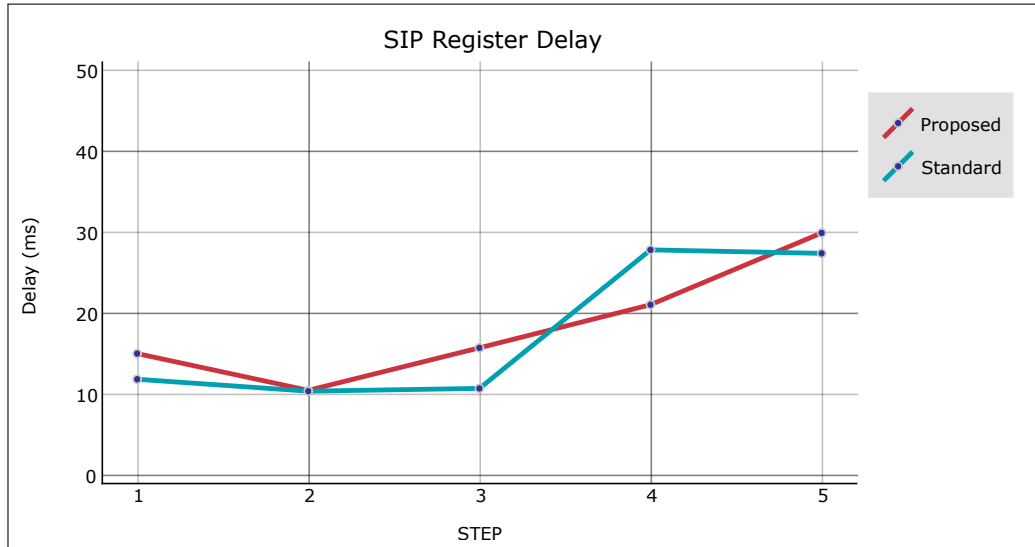


Figure 7.15: Mean delay during first registration exchange.

7.2.2 Standard call setup delay

A call was made between two users registered with the network to analyse the session setup delay. In the original scenario over the standardised architecture, the call is routed from the S-CSCF to the end user without invoking any ASs.

For the test over the proposed architecture the requests were redirected to the SCIM. Here they were evaluated against the configured rule set to determine if there was a need for any actions to be taken. The rule set for this particular scenario did not change the routing, and merely passed on the request according to normal SIP routing rules. It shows the minimum impact on call setup delay from the added hop in the signalling path. As there is now a new node in the SIP signalling path an increase in the overall call setup delay is expected.

Configuration

For this test, a preamble phase was established that consisted of registering enough users for the next phases of the test. To make sure that there were always enough users to sustain multiple simultaneous sessions a high number of users were registered. This ensured that there was always an available user configured in the user pools out of which the test could choose new users when beginning new sessions.

We registered users at a rate of 20 Calls Per Second (CPS) with a constant distribution until the number of registered users reached 5000. A stir phase was then conducted. This consisted of a mix of registration, de-registration and re-registration scenarios executed at 5 CPS. The duration of this phase was set to 5 minutes. The call distribution was configured to use a Poisson distribution as this is widely used to model call arrival rates in telecommunication literature. The percentage mix of the scenarios during the stir phase was as follows: Registration 40 %, De-registration 40 % and Re-registration 20 %.

The benchmark run phase consisted of only one scenario, namely the `ims_uac` scenario. This involved IMS bench spinning up a UE for two registered users and performing a call between them. IMS bench handles each side of this scenario, and uses the matching `ims_uas` scenario to handle the receiving end of the call.

As these scenarios were more involved, the overall load placed on the test bed in terms of number of CPS was reduced. These calls were executed starting at 1 CPS rising in steps of 1 until we hit 5 CPS. Each step was held constant for 60 seconds, and the results were averaged per step.

Load (CPS)	Mean (ms)	Minimum (ms)	90th Percentile
1	6.08	3.95	5.1
2	5.56	3.61	5.1
3	6.16	3.49	5
4	6.32	3.55	5.1
5	4.67	3.42	4.8

Table 7.7: Delay between INVITE sent and received over standard architecture.

Load (CPS)	Mean (ms)	Minimum (ms)	90th Percentile
1	7.4	6.12	7.8
2	7.72	5.97	8.5
3	8.74	6.58	9.7
4	10.51	7.49	11.5
5	12.88	8.35	14.5

Table 7.8: Delay between INVITE sent and received over proposed architecture.

Results

This test was repeated with or without the proposed architecture in place. However, for this scenario the intention was to ensure that, in the proposed architecture, the SCIM was actually placed in the SIP signalling path for the ensuing session. Thus it directly affected the session establishment. As before, the CPU and memory usage of the two architectures did not differ significantly. To determine how much delay the SCIM introduced, the time between the caller sending the SIP INVITE message and the callee receiving the SIP INVITE message was measured.

As can be seen in Table 7.7 and Table 7.8 there has been an increase in the session setup delay. The first message sent during the session establishment phase now has a minimum delay that has increased by roughly 2 milliseconds due to the extra hop (the SCIM) being present in the signalling path. This difference can be seen in Figure 7.16.

Load (CPS)	Mean (ms)	Minimum (ms)	90th Percentile
1	6.08	3.95	5.1
2	5.56	3.61	5.1
3	6.16	3.49	5.0
4	6.32	3.55	5.1
5	4.67	3.42	4.8

Table 7.9: Total session delay over standard architecture.

Load (CPS)	Mean (ms)	Minimum (ms)	90th Percentile
1	16.34	13.83	17.2
2	16.63	13.81	18.3
3	18.58	14.3	20
4	19.75	15.34	21.6
5	22.18	16.38	26.6

Table 7.10: Total session delay over proposed architecture.

This is expected as the SCIM is now in the signalling path of the incoming SIP messages, as can be seen in the signal diagrams shown previously. A small increase in delay is the result of processing each message together with the latency introduced by having another distinct server in the path. An increase of roughly 2 ms per message is well within tolerances for establishing sessions.

The overall session setup delay for the new architecture can be seen in Table 7.10. The increase in minimum session setup time is around 10 ms due to the higher latency per message sent during session setup.

Discussion

While the increased latency does increase the session setup delay, the overall session setup delay is expected to be at least two orders of magnitude larger than this.

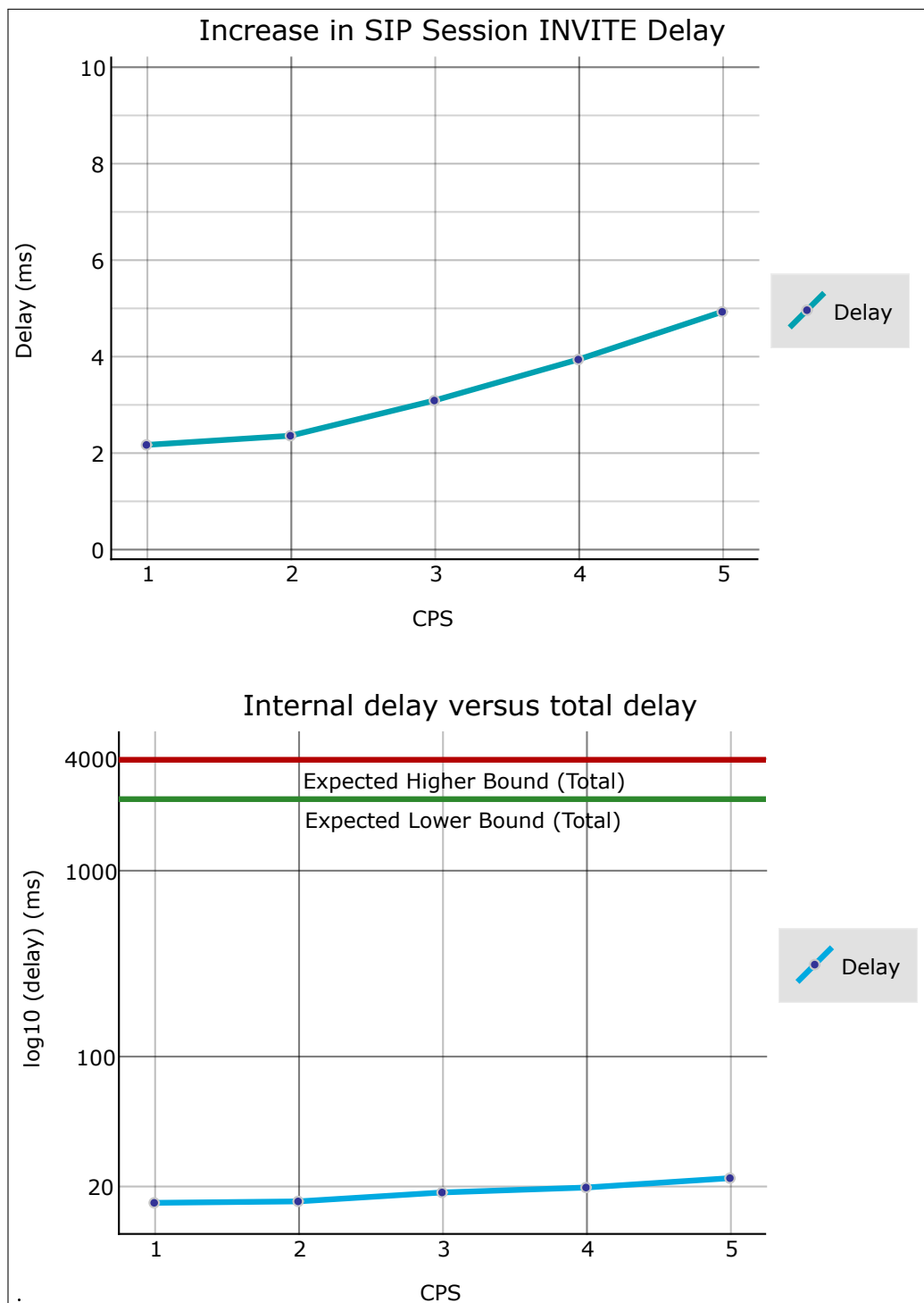


Figure 7.16: Comparison of session establishment delay.

Results for the average session setup delay in the IMS can be found in the literature and are between 2426 and 3950 milliseconds [108, 135, 136], reaching up to 8000 milliseconds for long distance sessions over GSM networks [137].

The additional delay here is less than 1 % of the expected total delay, as can be seen in Figure 7.16. As the number of services used during any one particular session increases, the significance of the delay due to the SCIM decreases. Thus for this comparison we considered the least favourable scenario, namely one where there are no additional services used during the session.

While our results do not take latency in the access network into account and thus show a much lower session setup delay, they show that the introduced delay inside the core network is marginal compared to the overall expected delay. The additional functionality can be added without incurring a significant penalty to the session establishment delay.

It is important to note that the single threaded nature of the implementation, together with the fact that it lacks a concurrent access database starts to influence the overall session delay reported here. As the proof of concept implementation was not expected to be performant, the focus is on the minimum increase in session setup delay as this represents the proposed architecture and not the implementation of the architecture. The increase in delay added by the proof of concept SCIM as load increases is shown in Figure 7.17.

This is expected as the SCIM needs to examine the incoming SIP messages which have increased due to the increase in the CPS. Each call actually represents multiple different SIP messages, as can be seen in the signal diagrams in the previous section.

It is also impacted by certain code artifacts in the proof of concept implementation, such as a single list of transactions which is protected from multiple simultaneous accesses through basic lock mechanisms. These lock mechanisms are known for decreasing throughput as load increases [138].

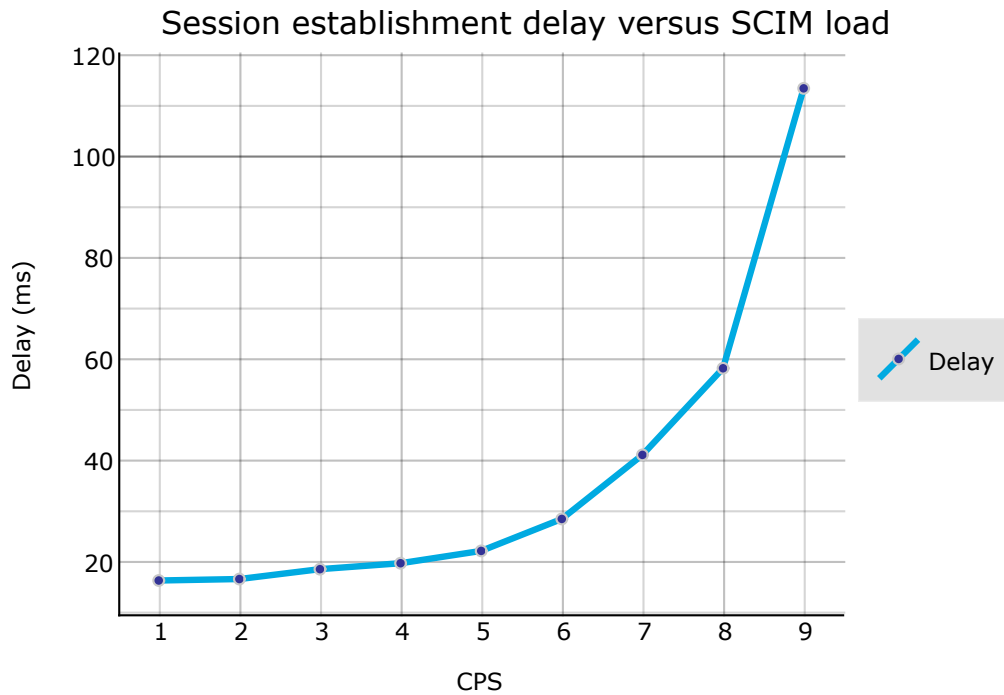


Figure 7.17: Increase in session setup delay with increased load.

7.2.3 Discussion

The proposed architecture has not increased the session establishment delay beyond acceptable bounds, even though the SCIM implementation is not performant. The framework integrates with existing testing tools without any modification, indicating a good level of backwards compatibility. The tests show that the proposed architecture was implemented successfully and demonstrates the viability of the additional functionality. The thesis is completed in the following chapter, which details several conclusions drawn from this work as well as recommendations for future work.

Chapter 8

Conclusions and Recommendations

This chapter discusses the conclusions drawn from this work and acts as a summary with the overall findings. It also outlines further areas of research that can be continued after this work.

The current approach to service invocation methods in the IMS as identified both by the standards bodies as well as related work in this area have been discussed. The thesis proposes specific requirements that should be met for any proposed solution. A new potential architecture was outlined to overcome some of the identified challenges, while remaining within the discussed requirements. The architecture was tested through an open source IMS test-bed utilising a public cloud infrastructure provider, and the related software was released as Open Source Software (OSS) to further research in this area.

The proof of concept implementation shows that it is feasible to introduce the architectural changes to existing IMS deployments without modifying core elements. These architectural changes introduce new functionality and increase the flexibility of the service triggering architecture. Previously impossible service usage scenarios can now be completed successfully. The

generic framework provided acts as a base for further work exploring more complex forms of service conflict detection and resolution mechanisms.

8.1 Conclusions

The main conclusions resulting from the work contained in this thesis are highlighted below.

8.1.1 IP based service competition

As can be seen in the first chapter of this thesis there has been an explosion of user focused services on the internet. This has been made possible by a huge increase in the bandwidth available to, and number of, end user devices.

It was found that traditional telecommunication operators face stiff competition from unexpected sources, including hardware manufacturers and new internet based service providers. The network operators monopoly on the end user's device and telecommunication service is at an end due to freely available VoIP and messaging clients. These clients have many millions of users across the world, sending billions of messages as well as using billions of minutes.

However, it was also determined that the IMS platform was well positioned to take advantage of this growth in device capability and network bandwidth, and has been instrumental in handling the evolution of telecommunication networks to packet based networks. This has allowed the network operator to focus on delivering rich, enhanced services in a more cost effective manner while decreasing the time taken to develop these services. This has in turn opened up several different paths for the network operator to compete and remain relevant in this new landscape.

8.1.2 User Centric Computing

By now users have become fully committed to the idea that their online spaces are customised to their needs. Not only do these spaces serve customised feeds of personalised content, they are also filled with user generated content.

This now accounts for a significant amount of the bandwidth used over the internet. It has been driven largely by the rise of the social networking spaces, and has helped to diversify the source of content providers. While this makes it more challenging to capture content revenue, it allows the telecommunication operator to act as a facilitator of this communication, profiting from the end user sharing their own content.

To fully take advantage of this trend the network operators need to look closely at allowing the end user to leverage their social network when utilising services as well as providing them with relevant, customised content. This is made easier by the IMS platform, as it allows a greater integration of different services than previously possible, by sharing a common underlying platform with building blocks that can be reused by many different services.

8.1.3 Standardised architecture

The standardised architecture in the IMS was built upon a well understood, open source protocol. The breakdown of common functionality into reusable sections such as Authentication or Authorisation, together with the separation of media and control plane signalling has allowed the rapid development of new, enriched services. While this architecture has worked out well, the 3GPP identified areas of deficiency that still needed to be tackled.

It was highlighted that there is currently no way for the end user to customise his or her service usage, nor use contextual information in routing decisions. There was a lack of conflict detection and resolution mechanisms, as well as a lack of more advanced error handling. The need for dynamic service invoca-

tion based on either current context or history of previous service invocations was raised, and it was determined that this needed to be undertaken in a backwards compatible way with minimum changes to existing architecture or deployments. Several different architecture types were presented as options for the SCIM topology, with the centralised SCIM model chosen based upon recommendations made by 3GPP.

8.1.4 State of art

The overall concept and motivation for this work was highlighted in a study commissioned by 3GPP, the standards body responsible for the IMS. As such, the literature review found that there has been a fair amount of research in the different areas that this thesis covers, such as:

- Service conflict detection and resolution
- Predetermining static service combinations
- Reducing the load on the S-CSCF
- Reducing the session setup delay
- Tracking session state

However there are still open areas revolving around:

- Automatic gathering of service information that can be used for routing decisions
- User customisation of routing decisions
- Contextual service routing
- IMS specific customisation of related work based on experiences with previous network standards

Several common themes were identified throughout the related literature, and these formed the basis of the requirements that are outlined in the next section.

8.1.5 Requirements for SCIM architecture

The review found that there are already substantial IMS deployments in the wild. Thus any proposed modifications to the core architecture should be backwards compatible. As one of the reasons behind the IMS was to increase the range and diversity of services offered to the end user, any changes should cater for this and not artificially limit the flexibility of the current system.

The system should also allow the end user a form of customisation or personalisation, allowing them to have some say in how their services are invoked or triggered. Finally, any proposed solution should aim to remain as efficient as possible and avoid any redundant signalling between individual AS and elements within the core network.

The work presented in this thesis integrates fully with the existing core elements in the IMS, allowing the network operator to determine the level of integration desired with their current deployment. They can choose various integration strategies, such as only moving particular users over or only invoking the new architecture for certain services. This is driven by the currently existing iFC architecture.

The proposed architecture also requires almost no changes to existing SIP ASs. It introduces no new interfaces, no new extensions to the SIP protocol as well as no new protocols. The only proposed modification of the SIP AS is the addition of a SIP MESSAGE which conveys information about the SIP AS. This can be replaced by manually conveying this information should the automatic service discovery and registry not be desirable. The IMS client and end user device do not require any modifications, as the interface between the end user and the USPS relies on the use of the standard HTTP protocol.

8.1.6 New functionality developed

By analysing the current state of art together with the proposed requirements from the standards body and the stated goals in this thesis, it was determined that new functionality was needed. The new functionality developed can be classified into four main sections:

- Conflict Detection and Resolution
- Service Registration
- User Controlled Customisation
- Contextual Routing

The system designed and developed in this thesis provides a new hybrid form of conflict detection and resolution. Normally these systems are classified into either offline or online conflict detection and resolution categories, as well as either static or dynamic systems. The proposed system contains features that are normally considered distinct to each category, and as such introduces a new flexible interaction management system to the IMS.

It also builds contextual routing into the core of the IMS network, allowing a much greater degree of routing flexibility as well as making contextual information available during service request routing. This allows dynamic service routing to take place, as routing decisions can be made depending on the context of the service or end user.

In order to achieve the dynamic service routing, a new information gathering system (including service discovery, registration and contextual information about the end user) was proposed and developed. This provides the necessary context to make more intelligent routing decisions as well as informing the end user about the available services so they can personalise or customise their service usage. The individual areas are discussed in more depth in the following sections.

8.1.7 Conflict Detection

Currently there are no specific, standardised conflict detection and resolution mechanisms in the IMS. As mentioned above, the proposed architecture in this thesis includes functionality aimed at this deficiency. The ability of the USPS to prevent the end user from choosing conflicting services is one such mechanism. This is primary an offline interaction management system, as conflicting service usage is prevented before the actual session establishment.

However, the SCIM is also capable of online interaction management as it is in the signalling path and has contextual information about the services being invoked. It is also a dynamic form of service conflict detection and resolution as it allows one to specify service usage or actions to be taken that depend on the result of previous operations. In other words it is possible to influence the routing of service requests according to the context of the system or user during the service execution. It is also possible to use the system in a static manner, where all the inputs are known ahead of time and do not change.

8.1.8 Service Registration

A new system that allows new services to register themselves with the IMS framework was developed. This allows new services (SIP ASs) to automatically supply information about their capabilities. This not only allows a list of available services to be populated, but also allows the relationship between different services to be mapped out. An automatic system to handle this is needed as the number of services is expected to raise rapidly, and manually inputting this information becomes infeasible. This mechanism also allows other information to be conveyed, such as service usage metrics. This aids the network operator in monitoring the behaviour and overall utilisation of the various services. The information about the services and their capabilities is also provided to the end user, which allows them to make informed

decisions about how they want their service invocation to be handled.

This information gathering was based upon standard IMS features such as the SIP signalling protocol and was modelled on the pre-existing presence functionality.

A new service description document type was proposed. This is an XML document which allows services to be described, and is based upon the Presence Information Data Format (PIDF). It is an extension to this format in the same vein as the SIP User Agent Capability Extension except aimed at IMS services rather than end user devices.

8.1.9 User Controlled Customisation

The work described in this thesis was one of the earliest forms of user customisable service routing as part of the core network in the IMS. The HTTP interface to the USPS allows the end user to select from a range of services, determining under what conditions each service is triggered, and how they should be linked together. This is possible as it has information about the available services due to the service registration mechanism discussed above. It also has information about possible conditions or contextual information that is available to be used, allowing the end user to create a custom service usage scenario that includes multiple services and conditions.

Once the end user has created a particular set of service interactions they would like to utilise, these are stored and transmitted to the SCIM as a defined rule-set. This rule-set is consulted whenever the SCIM receives a SIP message, allowing it to influence the SIP session as needed, activating new services or re-routing SIP messages. The rule-sets are defined per user, but it is envisaged that the network operator would have a set of pre-defined rule-sets that are available for further customisation should any one particular user be interested in changing the defaults.

8.1.10 Contextual Routing

Previously, SIP routing in the IMS was controlled through the iFC. These had a very limited, statically defined set of trigger points that they could utilise to determine how to route messages.

The architecture in this thesis allows these decisions to be made not only by these trigger points, but also on the context of the end user. The context of the end user could contain user specific items such as their current presence status or location, or more general context such as the time of day. This was done without requiring any modifications to the S-CSCF, nor any custom extensions to SIP. It does not modify the individual SIP messages by adding additional parameters, and allows actions to be taken on any SIP message, not only the initial request.

This allows the routing decision to be influence at various stages of the session, and not just on the initial request. This contextual routing is available to be used should it be desirable, and it is possible for both routing types to co-exist. The network operator can use the standard iFC to control whether or not the more advanced routing should be used.

8.1.11 Cloud based test-bed

The open-source, cloud based test-bed was invaluable to investigate the proposed architecture. The test-bed allowed a proof of concept implementation to be developed and evaluated in a practical setting. Without the open source nature of the Fraunhofer FOKUS Open Source IMS Core this work would not have been possible. All components developed during this thesis have also been released as open source software, making this work accessible to future researchers.

Not only was the test-bed used extensively during this work, it is also envisaged that it makes it easier for future researchers to continue this work

as it can be replicated without having to invest a fair amount of capital in deploying physical machines. However, it was noted that the performance of public cloud based infrastructure is not one hundred percent consistent, and as such is not ideal for benchmarking or performance testing. Despite the variable performance it is well suited to functional testing under a wide range of conditions.

8.1.12 Session Setup Delay

The minimum increase in session setup delay discovered during testing verifies that it is possible to add a single extra hop in the signalling path without incurring heavy penalties to the session setup delay. The increase is at least an order of magnitude less than the overall session setup delay.

We believe that the added functionality far outweighs this small increase in session setup delay, and further more the proposed architecture allows this delay increase to only apply to sessions requiring this new functionality. It is possible for normal sessions to not incur this penalty by bypassing the SCIM and being routed as normal. The involvement of the SCIM can be tailored through the use of iFC, which currently provide all manipulation of service routing decisions for the IMS.

The proof of concept implementation showed that special care must be given when developing a SIP server that fulfils such a critical role in the session establishment. It is possible for this delay to increase unbounded in line with an increasing call volume, rendering the session setup delay significant should the SCIM's performance not match those of the other existing CSCFs.

8.2 Future work

While the proposed architecture has met the objectives laid out at the start of this thesis, several areas of future work have been identified. These are discussed below.

8.2.1 Optimised implementation

The proof of concept prototype developed to evaluate the architecture has several limitations as a result of being developed during a research project. It has very verbose logging and a fair amount of debug information is present, including captures of all messages. Very primitive synchronisation mechanisms were used and, as revealed during testing, the delay introduced increases with call volume due to this contention.

A more optimised version could be developed to evaluate behaviour at very high call volumes and investigate how many SCIMs would be needed to handle certain call loads at an acceptable delay level. Such a SCIM should be based upon an entity with proven performance such as Kamailio (formally the SIP express router) [139] instead of being written from scratch.

Another avenue that could be investigated, with a more performant implementation, would be examining how the complexity of the configured rule-set affects the session establishment delay. Benchmarking the number of users that can be supported by a single SCIM and investigating its horizontal scalability is also left for further study. The benchmarking of the CSCFs in such a way was done by Bellavista et al. and should serve as a starting point for this investigation [140].

8.2.2 Service conflict resolution

The service conflict detection and resolution mechanism presented in the proof of concept prototype is fairly naive. It relies on a small set of basic heuristics, such as the modification of the same header by one or more services, to determine whether or not there is a conflict.

In addition, the resolution mechanism is pre-emptive. That is, it does not take place during or after a conflicting set of services have been activated, but instead attempts to prevent the problematic service interaction from happening in the first place. The detection mechanism should be expanded to provide more parameters and types of heuristics, including results of previous service invocation or allowing the network operator to manually disable service interactions that are undesirable for some reason (e.g. potential market segmentation or billing purposes).

While this thesis has focused primarily on service conflict detection and resolution in the IMS space, this is a common problem in many more fields. A similar feature interaction problem exists in home automation systems . It would be interesting to see how closely findings from this area could be applied to the telecommunication space. For example, Pedersen et al. apply model checking as analysis tool to investigate feature interactions between disparate systems [141].

8.2.3 Machine learning

The current approach relies on manual interaction by the end user to create customised service usage scenarios. However, these service combinations are likely to be useful to other end users as well. Future work involves automatically classifying service usage scenarios based on captured sessions and generating templates to be presented to the end user. Previously machine learning has been used to automatically classify dangerous SIP messages in the IMS [142].

A similar approach could be used but targetted to classify services instead. This can also be applied to the SCIM's intelligence for handling unforeseen service conflicts, allowing it to choose a sane approach when it encounters an unknown interaction. As the number of services increase and the interactions grow more complex, the need for machine guided algorithms will become increasingly apparent.

References

- [1] Apple Press Release, “Apple’s App Store Marks Historic 50 Billionth Download,” May 2013.
[Online]. Available: <http://www.apple.com/pr/library/2013/05/16Apples-App-Store-Marks-Historic-50-Billionth-Download.html>
- [2] M. Meeker and L. Wu, “Internet Trends 2013.” DD 11 Conference, May 2013.
[Online]. Available:
<http://www.kpcb.com/insights/2013-internet-trends>
- [3] J. Poushter, “Smartphone Ownership and Internet Usage Continues to Climb in Emerging Economies,” Feb. 2016.
[Online]. Available: <http://www.pewglobal.org/2016/02/22/smartphone-ownership-and-internet-usage-continues-to-climb-in-emerging-economies/>
- [4] Alan Quayle Business and Service Development, “IMS Status Report: Moving up the slope of enlightenment,” March 2012.
[Online]. Available:
<http://www.infonetics.com/pr/2009/4q08-ims-market-highlights.asp>
- [5] TeleGeography, “Executive Summary: TeleGeography Report,” February 2013.
[Online]. Available: <http://www.telegeography.com/press/press-releases/2013/02/13/the-bell-tolls-for-telcos/index.html>
- [6] Informa, “VoIP and IP messaging: Operator strategies to combat the threat from OTT players,” May 2013.

- [Online]. Available: <https://commerce.informatm.com/reports/main/voip-ip-messaging-revised.html>
- [7] Microsoft Investor Relations, “Earnings Release FY13 Q1.”
[Online]. Available:
<http://www.microsoft.com/investor/EarningsAndFinancials/Earnings/PressReleaseAndWebcast/FY13/Q1/default.aspx>
- [8] Skype Blog, “Thanks for Making Skype A Part of Your Daily Lives.”
[Online]. Available: <http://blogs.skype.com/2013/04/03/thanks-for-making-skype-a-part-of-your-daily-lives-2-billion-minutes-a-day/>
- [9] “2015: Skype’s year in review,” Dec. 2015.
[Online]. Available:
<http://blogs.skype.com/2015/12/17/2015-skypes-year-in-review/>
- [10] “Viber - Profile,” May 2013.
[Online]. Available: <http://www.viber.com/about>
- [11] Rakuten, “FY2014 Second Quarter Consolidated Financial Results.”
[Online]. Available: http://global.rakuten.com/corp/investors/documents/pdf/14Q2PPT_E.pdf
- [12] CNET, “Twilio takes on the world with its cloud-based telephony.”
[Online]. Available: http://news.cnet.com/8301-1023_3-57534234-93/twilio-takes-on-the-world-with-its-cloud-based-telephony/
- [13] Twilio Inc., “SEC Filing, Form S1.”
[Online]. Available: <https://www.sec.gov/Archives/edgar/data/1447669/000104746916013448/a2227414zs-1.htm>
- [14] Facebook Inc., “Transcript of First Quarter 2016 Results Conference Call.”
[Online]. Available: https://s21.q4cdn.com/399680738/files/doc_financials/2016/FB_Q116_Transcript.pdf

- [15] 2Go Press Release, “2go launches profile previews in chat rooms,” August 2015.
[Online]. Available:
http://www.2go.im/press/201508-2go_adds_room_profile_previews
- [16] “Facebook Reports First Quarter 2016 Results and Announces Proposal for New Class of Stock.”
[Online]. Available: https://s21.q4cdn.com/399680738/files/doc_news/2016/FB_News_2016_4_27_Financial_Releases.pdf
- [17] Facebook Press Release, “Thank You Messenger.”
[Online]. Available:
<https://newsroom.fb.com/news/2016/07/thank-you-messenger/>
- [18] LeadLedger, “Latest Facebook Connect usage statistics, client lists and other social marketshare information.”
[Online]. Available:
<http://www.leadledger.com/tech/Facebook-Connect>
- [19] Facebook, “Form 10-K - Annual Report,” February 2013.
[Online]. Available:
<http://investor.fb.com/secfiling.cfm?filingID=1326801-13-3>
- [20] VidCon, “Industry Keynote with YouTube CEO Susan Wojcicki (VidCon 2015).”
[Online]. Available:
<https://www.youtube.com/watch?v=O6JPxCBlBh8>
- [21] “Google Voice - Android Apps on Google Play.”
[Online]. Available: <https://play.google.com/store/apps/details?id=com.google.android.apps.googlevoice&hl=en>
- [22] J. Bannister, P. Mather, and S. Coope, *Convergence Technologies for 3G Networks*. John Wiley & Sons, 2004.
- [23] ITU-T Recommendation Y.2001, “General overview of NGN,” Dec. 2004.

- [Online]. Available: <http://www.itu.int/ITU-T/ngn/definition.html>
- [24] 3GPP TS 23.002 V12.1.0, “Network Architecture; Release 12,” Dec. 2012.
- [25] A. Sanchez-Esguevillas, B. Carro, G. Camarillo, Y.-B. Lin, M. Garcia-Martin, and L. Hanzo, “IMS: the new generation of internet-protocol-based multimedia services,” *Proceedings of the IEEE*, vol. 101, no. 8, pp. 1860–1881, 2013.
- [26] 3GPP TS 23.218 V12.2.0, “IP Multimedia (IM) session handling; IM call model; Stage 2; Release 12,” Jun. 2013.
- [27] A. Lee, “Application creation for IMS systems through macro-enablers and web 2.0 technologies,” *Bell Labs Technical Journal*, vol. 15, no. 1, pp. 23–51, 2010.
[Online]. Available: <http://onlinelibrary.wiley.com/doi/10.1002/bltj.20422/abstract>
- [28] Global mobile Suppliers Association, “Evolution to LTE report: 480 LTE networks launched in 157 countries,” Oct. 2015.
[Online]. Available: http://gsacom.com/wp-content/uploads/2015/10/151013-Evolution_to_LTE_report.pdf
- [29] 3GPP TS 23.228 V12.1.0, “IP Multimedia Subsystem (IMS); Stage 2; Release 12,” Jun. 2013.
- [30] Open Mobile Alliance, “Presence Simple Version 2.0. Candidate Enabler Release,” Dec. 2010.
- [31] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, “SIP: Session Initiation Protocol,” RFC 3261, Internet Engineering Task Force, June 2002, updated by RFCs 3265, 3853, 4320, 4916, 5393.
[Online]. Available: <http://www.ietf.org/rfc/rfc3261.txt>

- [32] 3GPP TS 24.229 V12.1.0, “IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3; Release 12,” Jun. 2013.
- [33] J. Postel, “User Datagram Protocol,” RFC 768, Aug. 1980.
[Online]. Available: <http://www.ietf.org/rfc/rfc768.txt>
- [34] J. Postel, “Transmission Control Protocol,” RFC 793, Internet Engineering Task Force, Sep. 1981, updated by RFCs 1122, 3168, 6093, 6528.
[Online]. Available: <http://www.ietf.org/rfc/rfc793.txt>
- [35] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications,” RFC 3550, Internet Engineering Task Force, Jul. 2003, updated by RFCs 5506, 5761, 6051, 6222.
[Online]. Available: <http://www.ietf.org/rfc/rfc3550.txt>
- [36] V. Fajardo, J. Arkko, J. Loughney, and G. Zorn, “Diameter Base Protocol,” RFC 6733 (Proposed Standard), Internet Engineering Task Force, Oct. 2012.
[Online]. Available: <http://www.ietf.org/rfc/rfc6733.txt>
- [37] 3GPP TS 29.229 V12.0.0, “Cx and Dx interfaces based on the Diameter protocol; Protocol details; Release 12,” Jun. 2013.
- [38] A. Gouya and N. Crespi, “Service Orchestration in IMS,” *IMS Handbook - Concepts, Technologies and Services - Mohamed Ilyas and Syad Ahson*, pp. 329–344, Nov. 2008.
- [39] L. Zhou, L. H. Ngoh, T. Y. Chai, T. K. Lee, X. Shao, and J. C. M. Teo, “Design and Implementation of an IMS-Based Testbed for Real-Time Services Orchestration and Delivery in Heterogeneous Networks,” in *Service Delivery Platforms*. Auerbach Publications, Mar. 2011, pp. 195–212.
[Online]. Available: <https://doi.org/10.1201/b10784-10>

- [40] R. Spiers, R. Good, and N. Ventura, "Service Orchestration in the IP Multimedia Subsystem," in *Service Delivery Platforms*. Auerbach Publications, Mar. 2011, pp. 175–193.
[Online]. Available: <http://dx.doi.org/10.1201/b10784-9>
- [41] 3GPP TS 22.228 V12.6.0, "Service requirements for the Internet Protocol (IP) Multimedia core network Subsystem (IMS); Stage 1 ; Release 12," Jun. 2013.
- [42] M. Calder, M. Kolberg, E. Magill, and S. Reiff-Marganiec, "Feature interaction: a critical review and considered forecast," *Computer Networks*, vol. 41, no. 1, pp. 115–141, January 2003.
[Online]. Available: <http://eprints.gla.ac.uk/2874/>
- [43] A. Gouya and N. Crespi, "Detection and resolution of feature interactions in IP multimedia subsystem," *International Journal of Network Management*, vol. 19, no. 4, pp. 315–337, 2009.
[Online]. Available: <http://onlinelibrary.wiley.com/doi/10.1002/nem.710/abstract>
- [44] N. Crespi, "A distributed mechanism to resolve dynamically Feature Interaction in the UMTS IP Multimedia Subsystem," in *Sixth International Workshop on Applications and Services in Wireless Networks (ASWN)*, Berlin, May 2006.
- [45] Z. Chentouf, S. Cherkaoui, and A. Khoumsi, "Implementing online feature interaction detection in SIP environment: early results," in *10th International Conference on Telecommunications, 2003. ICT 2003*, vol. 1, 2003, pp. 515–521 vol.1.
- [46] 3GPP, "23.810 v8.0.0, On Architecture Impacts of Service Brokering," (*Release 8*)(*SA2*), Sep. 2008.
- [47] J. Rosenberg and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)," RFC 3264 (Proposed Standard), Internet Engineering Task Force, Jun. 2002, updated by RFC 6157.

- [Online]. Available: <http://www.ietf.org/rfc/rfc3264.txt>
- [48] P. Rohan, R. Sunku, P. Airan, M. Pande, and D. Das, “IMS service broker SCIM enriching REST based web 2.0 mashup’s,” in *2010 IEEE 4th International Conference on Internet Multimedia Services Architecture and Application(IMSAA)*, Dec. 2010, pp. 1 –6.
 - [49] 3GPP TS 23.218 V0.6.0, “IP Multimedia (IM) session handling; IM call model; Release 5,” Sep. 2001.
 - [50] K.-D. Chang, C.-Y. Chen, J.-L. Chen, and H.-C. Chao, “Challenges to next generation services in ip multimedia subsystem,” *Journal of Information Processing Systems*, vol. 6, no. 2, pp. 129–146, 2010.
 - [51] “3GPP TS 22.127 v7.1.0, "Service requirement for the open services access (OSA); stage 1; release7," Mar. 2006.
 - [52] M. Kolberg and E. H. Magill, “Managing feature interactions between distributed sip call control services,” *Computer Networks*, vol. 51, no. 2, pp. 536 – 557, 2007.
[Online]. Available: <http://www.sciencedirect.com/science/article/B6VRG-4KXVB79-1/2/d2ba17665d6f030e94ce37a2444efc63>
 - [53] T. Berners-Lee, R. Fielding, and L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax,” RFC 3986, Internet Engineering Task Force, Jan. 2005, updated by RFC 6874.
[Online]. Available: <http://www.ietf.org/rfc/rfc3986.txt>
 - [54] Open Mobile Alliance, “Enabler Release Definition for XML Document Management,” Apr. 2012.
 - [55] 3GPP TS 23.002 V12.2.0, “Network architecture (Release 12),” Jun. 2013.
 - [56] 3GPP TS 29.228 V12.0.0, “IP Multimedia (IM) Subsystem Cx and Dx interfaces; Signalling flows and message contents (Release 12),” Sep. 2013.

- [57] A. Gouya, N. Crespi, and E. Bertin, "Service invocation issues within the IP multimedia subsystem," in *International conference on Networking and Services (ICNS'06)*. Los Alamitos, CA, USA: IEEE Computer Society, 2007.
- [58] Chua, Tan, and Ho, "A study of issues and considerations of service interaction management in IMS architecture," *WSEAS Transactions on Computers*, vol. 7, no. 9, p. 1405–1415, 2008.
- [59] J. Liao, Q. Qi, Z. Xun, T. Li, Y. Cao, and J. Wang, "A linear chained approach for service invocation in IP multimedia subsystem," *Computers & Electrical Engineering*, vol. 38, no. 4, pp. 840–852, Jul. 2012.
[Online]. Available: <http://www.sciencedirect.com/science/article/pii/S004579061200050X>
- [60] R. Goveas, R. Sunku, and D. Das, "Centralized Service Capability Interaction Manager (SCIM) architecture to support dynamic-blended services in IMS network," in *Proc. 2nd International Conference on Internet Multimedia Services Architecture and Applications IMSAA 2008*, Dec. 2008, p. 1–5.
- [61] E. Guttman, C. Perkins, J. Veizades, and M. Day, "Service Location Protocol, Version 2," RFC 2608 (Proposed Standard), Internet Engineering Task Force, Jun. 1999, updated by RFC 3224.
[Online]. Available: <http://www.ietf.org/rfc/rfc2608.txt>
- [62] J. Sun and X. Li, "An analytical model for centralized Service Discovery architecture in IMS networks," in *IEEE Youth Conference on Information, Computing and Telecommunication, 2009. YC-ICT '09*, Sep. 2009, pp. 251–254.
- [63] J. Zhang, "Service Discovery Architecture Based on IMS for Future NGN," in *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*, Oct. 2008, pp. 1–5.

- [64] W. Chiang and C. Tseng, "Handling feature interactions for multi-party services in 3GPP IP multimedia subsystem," in *Proc. International Conference on IP Multimedia Subsystem Architecture and Applications*, Dec 2007, p. 1–5.
- [65] J. Xu, X. Wei, and C. Fan, "Experiential learning-based feature interaction detection in IMS," in *2010 3rd IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT)*, 2010, pp. 823–827.
- [66] L. Hua, Z. Xue, and Y. Fangchun, "Unified service conflict control in IMS," in *10th International Conference on Telecommunications, 2009. ICT 2009*, 2009, pp. 397–404.
- [67] A. Gouya and N. Crespi, "Service broker for managing feature interactions in IP multimedia subsystem," in *Sixth International Conference on Networking, 2007. ICN '07*, 2007, pp. 54–54.
- [68] R. Goveas, R. Sunku, M. Pande, and D. Das, "Workflow based design for blending services over IP multimedia subsystems (IMS) with service capability interaction manager," in *Internet Multimedia Services Architecture and Applications (IMSAA), 2009 IEEE International Conference on*, 2009, pp. 1–5.
- [69] M. Nakajima, M. Kaneko, M. Hirano, and H. Sunaga, "SIP servlet dialogue handling for NGN Service Capability Interaction Manager," in *Proc. APSITT Information and Telecommunication Technologies 7th Asia-Pacific Symposium on*, Apr. 2008, p. 41–46.
- [70] M. Varatanovic, N. Secic-Haracic, M. Skrbic, M. Hadzialic, and E. Grabovica, "Overview of IMS application layer interaction management," in *2012 IX International Symposium on Telecommunications*, 2012, pp. 1–5.
- [71] J. Zha, J. Wang, R. Han, and M. Song, "Research on load balance of service capability interaction management," in *2010 3rd IEEE*

International Conference on Broadband Network and Multimedia Technology (IC-BNMT), Oct. 2010, pp. 212–217.

- [72] W. Fangyi, L. Xia, and Z. Hua, “SSTM: a state based IMS service triggering mechanism,” in *Communications and Mobile Computing (CMC), 2010 International Conference on*, vol. 1, 2010, pp. 508–512.
- [73] M. Ulvan and R. Bestak, “Delay performance of session establishment signaling in IP multimedia subsystem,” in *16th International Conference on Systems, Signals and Image Processing, 2009. IWSSIP 2009*, 2009, pp. 1–5.
- [74] Z. Xun, J. Liao, and X. Zhu, “On performance of 3GPP service triggering mechanism in IMS network,” in *Software Engineering and Advanced Applications, 2008. SEAA '08. 34th Euromicro Conference*, 2008, pp. 150–155.
- [75] Z. Xun, J. Liao, X. Zhu, C. Wang, and Y. Cao, “A group based service triggering algorithm for IMS network,” in *IEEE International Conference on Communications, 2009. ICC '09*, 2009, pp. 1–6.
- [76] Q. Qi, J. Liao, X. Zhu, and Y. Cao, “DSCIM: A Novel Service Invocation Mechanism in IMS,” in *Proc. IEEE Global Telecommunications Conference IEEE GLOBECOM 2008*, Dec. 2008, p. 1–5.
- [77] S. Krishnamoorthy and J. Torres, “IMS enhanced filter and action criteria,” in *IP Multimedia Subsystem Architecture and Applications, 2007 International Conference on*, 2007, pp. 1–3.
- [78] J. Liao, Z. Xun, C. Wang, and X. Zhu, “A call-state-based service triggering algorithm for IMS network,” *International Journal of Communication Systems*, vol. 22, no. 3, pp. 343–364, 2009.
 [Online]. Available:
<http://onlinelibrary.wiley.com/doi/10.1002/dac.973/abstract>

- [79] W. Chiang and K. Chang, "Design and implementation of a distributed service invocation function for the IP multimedia subsystem," *Computer Communications*, vol. 34 Issue 9, June, 2011.
[Online]. Available:
<http://www.sciencedirect.com.ezproxy.uct.ac.za/science/article/B6TYP-4YFCF96-2/2/1ce3edbe70ac48d54ef4dfcf62f9fc95>
- [80] J. Rosenberg, J. Peterson, H. Schulzrinne, and G. Camarillo, "Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP)," RFC 3725 (Best Current Practice), Apr. 2004.
[Online]. Available: <http://www.ietf.org/rfc/rfc3725.txt>
- [81] L. Al-Doski, A. Abbosh, R. Babiceanu, and S. Mohan, "Optimizing resources utilization in IP multimedia subsystem," in *Systems Conference (SysCon), 2012 IEEE International*, 2012, pp. 1–5.
- [82] B. Zhu, "Analysis of SIP in UMTS IP multimedia subsystem," Ph.D. dissertation, North Carolina State University, Raleigh, 2003.
[Online]. Available:
<http://repository.lib.ncsu.edu/ir/bitstream/1840.16/2127/1/etd.pdf>
- [83] A. Acharya, N. Banerjee, D. Chakraborty, and S. Sharma, "Context-based applications in converged networks: Beyond SIMPLE presence," *IEEE Pervasive Computing*, vol. 12, no. 2, pp. 59–67, 2013.
- [84] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell, "Sensing meets mobile social networks: the design, implementation and evaluation of the CenceMe application," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, ser. SenSys 2008. New York, NY, USA: ACM, 2008, pp. 337–350.
[Online]. Available: <http://doi.acm.org/10.1145/1460412.1460445>
- [85] N. Banerjee, D. Chakraborty, K. Dasgupta, S. Mittal, S. Nagar, and Saguna, "R-u-in? - exploiting rich presence and converged

- communications for next-generation activity-oriented social networking,” in *Tenth International Conference on Mobile Data Management: Systems, Services and Middleware, 2009. MDM '09*, 2009, pp. 222–231.
- [86] H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell, “SoundSense: scalable sound sensing for people-centric applications on mobile phones,” in *Proceedings of the 7th international conference on Mobile systems, applications, and services*, ser. MobiSys '09. New York, NY, USA: ACM, 2009, pp. 165–178.
[Online]. Available: <http://doi.acm.org/10.1145/1555816.1555834>
- [87] E. Miluzzo, C. T. Cornelius, A. Ramaswamy, T. Choudhury, Z. Liu, and A. T. Campbell, “Darwin phones: the evolution of sensing and inference on mobile phones,” in *Proceedings of the 8th international conference on Mobile systems, applications, and services, Mobisys '10*. New York, NY, USA: ACM, 2010, pp. 5–20.
[Online]. Available: <http://doi.acm.org/10.1145/1814433.1814437>
- [88] J. Rosenberg, “A Presence Event Package for the Session Initiation Protocol (SIP),” RFC 3856 (Proposed Standard), Aug. 2004.
[Online]. Available: <http://www.ietf.org/rfc/rfc3856.txt>
- [89] A. Roach, “SIP-Specific Event Notification,” RFC 6665 (Proposed Standard), Jul. 2012.
[Online]. Available: <http://www.ietf.org/rfc/rfc6665.txt>
- [90] H. Khartabil, E. Leppanen, M. Lonnfors, and J. Costa-Requena, “Functional Description of Event Notification Filtering,” RFC 4660, Internet Engineering Task Force, Sep. 2006, updated by RFC 6665.
[Online]. Available: <http://www.ietf.org/rfc/rfc4660.txt>
- [91] H. Sugano, S. Fujimoto, G. Klyne, A. Bateman, W. Carr, and J. Peterson, “Presence Information Data Format (PIDF),” RFC 3863 (Proposed Standard), Aug. 2004.
[Online]. Available: <http://www.ietf.org/rfc/rfc3863.txt>

- [92] H. Schulzrinne, V. Gurbani, P. Kyzivat, and J. Rosenberg, "RPID: Rich Presence Extensions to the Presence Information Data Format (PIDF)," RFC 4480 (Proposed Standard), July 2006.
[Online]. Available: <http://www.ietf.org/rfc/rfc4480.txt>
- [93] H. Schulzrinne, "CIPID: Contact Information for the Presence Information Data Format," RFC 4482 (Proposed Standard), July 2006.
[Online]. Available: <http://www.ietf.org/rfc/rfc4482.txt>
- [94] H. Schulzrinne, "Timed Presence Extensions to the Presence Information Data Format (PIDF) to Indicate Status Information for Past and Future Time Intervals," RFC 4481 (Proposed Standard), Jul. 2006.
[Online]. Available: <http://www.ietf.org/rfc/rfc4481.txt>
- [95] G. Camarillo and M. Garcia-Martin, *The 3G IP Multimedia Subsystem (IMS): Merging the Internet and the Cellular Worlds*, 3rd ed. Wiley, Nov. 2008.
- [96] J. Rosenberg, "Presence Authorization Rules," RFC 5025 (Proposed Standard), Dec. 2007.
[Online]. Available: <http://www.ietf.org/rfc/rfc5025.txt>
- [97] J. Rosenberg, "The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)," RFC 4825 (Proposed Standard), May 2007.
[Online]. Available: <http://www.ietf.org/rfc/rfc4825.txt>
- [98] M. Isomaki and E. Leppanen, "An Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Usage for Manipulating Presence Document Contents," RFC 4827 (Proposed Standard), May 2007.
[Online]. Available: <http://www.ietf.org/rfc/rfc4827.txt>

- [99] A. B. Roach, B. Campbell, and J. Rosenberg, "A Session Initiation Protocol (SIP) Event Notification Extension for Resource Lists," RFC 4662 (Proposed Standard), Aug. 2006.
[Online]. Available: <http://www.ietf.org/rfc/rfc4662.txt>
- [100] A. Niemi, M. Lonnfors, and E. Leppanen, "Publication of Partial Presence Information," RFC 5264 (Proposed Standard), Sept. 2008.
[Online]. Available: <http://www.ietf.org/rfc/rfc5264.txt>
- [101] 3GPP, "3GPP TS 24.141 v12.0.0, "Presence service using the IP multimedia (IM) core network (CN) subsystem; stage 3; release 12" (2013)," (*Release 12*), Sep. 2013.
- [102] W. Zhou, J. Chang, and J. Song, "Research on context-awareness service adaptation mechanism in IMS under ubiquitous network," in *Vehicular Technology Conference (VTC Spring), 2012 IEEE 75th*, 2012, pp. 1–5.
- [103] R. Sunku, R. Pascal, and D. Das, "Intelligent offline charging for a blended mobile non-SIP gaming service in IMS network," in *2011 IEEE 5th International Conference on Internet Multimedia Systems Architecture and Application (IMSAA)*, Dec. 2011, pp. 1 –6.
- [104] M. Lonnfors and K. Kiss, "Session Initiation Protocol (SIP) User Agent Capability Extension to Presence Information Data Format (PIDF)," RFC 5196 (Proposed Standard), Sept. 2008.
[Online]. Available: <http://www.ietf.org/rfc/rfc5196.txt>
- [105] S. Pandey, V. Jain, D. Das, V. Planat, and R. Periannan, "Performance study of IMS signaling plane," in *2007 International Conference on IP Multimedia Subsystem Architecture and Applications*, 2007, pp. 1–5.
- [106] R Spiers, "Open Source IMS Research Implementations," January 2017.
[Online]. Available: <http://richardspiers.github.io/>

- [107] R. Spiers and N. Ventura, “An evaluation of architectures for ims based video conferencing,” in *Proceedings of the Southern African Telecommunication Networks and Applications Conference (SATNAC 2008)*, September 2008.
- [108] D. Waiting, R. Good, R. Spiers, and N. Ventura, “Open source development tools for IMS research,” *2008 4th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM’08)*, Mar. 2008.
- [109] D. Waiting, R. Good, R. Spiers, and N. Ventura, “The UCT IMS client,” in *5th International Conference on Testbeds and Research Infrastructures for the Development of Networks Communities and Workshops, 2009. TridentCom 2009*, 2009, pp. 1–6.
- [110] “Open IMS Core Mailing Lists,” February 2014.
[Online]. Available: <http://www.openimscore.org/maillists>
- [111] Mono Website, “About Mono,” February 2014.
[Online]. Available: <http://www.mono-project.com/About>
- [112] Inexbee, “Mercuro IMS Client,” Feb 2010.
[Online]. Available: <http://imsclient.blogspot.co.za/2010/02/new-release-of-mercuro-ims-client.html>
- [113] Doubango Telecom, “Boghe IMS/RCS client,” January 2017.
[Online]. Available: <https://github.com/DoubangoTelecom/boghe/>
- [114] Lumisoft, “Lumisoft .NET Class Reference,” February 2014.
[Online]. Available: <http://www.lumisoft.ee/lsWWW/download/downloads/Net/Help/Index.html>
- [115] D. Willis and B. Hoeneisen, “Session Initiation Protocol (SIP) Extension Header Field for Service Route Discovery During Registration,” RFC 3608 (Proposed Standard), Internet Engineering Task Force, Oct. 2003, updated by RFC 5630.

- [Online]. Available: <http://www.ietf.org/rfc/rfc3608.txt>
- [116] GStreamer, “GStreamer - Open Source Multimedia Framework,” January 2017.
[Online]. Available: <http://gstreamer.freedesktop.org/>
- [117] M. Bostock, “D3 - Data-Driven Documents,” January 2015.
[Online]. Available: <http://d3js.org/>
- [118] J. Mwangama, R. Spiers, N. Ventura, and T. Magedanz, “Past and current IMS testbed initiatives: The UCT IMS and EPC testbed,” in *2012 IEEE Globecom Workshops (GC Wkshps)*, Dec. 2012, pp. 1718–1723.
- [119] R. Glitho, “Cloudifying the 3gpp IP Multimedia Subsystem: Why and How?” in *2014 6th International Conference on New Technologies, Mobility and Security (NTMS)*, Mar. 2014, pp. 1–5.
- [120] P. Bellavista, G. Carella, L. Foschini, T. Magedanz, F. Schreiner, and K. Campowsky, “QoS-aware elastic cloud brokering for IMS infrastructures,” in *2012 IEEE Symposium on Computers and Communications (ISCC)*, Jul. 2012, pp. 157–160.
- [121] W. Zhang, W. Lei, X. Chen, and S. Liu, “Architecture and Key Issues of IMS-Based Cloud Computing,” in *2013 IEEE Sixth International Conference on Cloud Computing (CLOUD)*, Jun. 2013, pp. 629–635.
- [122] SmokePing Website, “About SmokePing,” June 2015.
[Online]. Available: <http://oss.oetiker.ch/smokeping/>
- [123] G. Wang and T. Ng, “The Impact of Virtualization on Network Performance of Amazon EC2 Data Center,” in *2010 Proceedings IEEE INFOCOM*, Mar. 2010, pp. 1–9.
- [124] A. Lenk, M. Menzel, J. Lipsky, S. Tai, and P. Offermann, “What Are You Paying For? Performance Benchmarking for Infrastructure-as-a-Service Offerings,” in *2011 IEEE International Conference on Cloud Computing (CLOUD)*, Jul. 2011, pp. 484–491.

- [125] T. Magedanz, D. Witaszek, and K. Knuettel, "The IMS playground @ fokus - an open testbed for next generation network multimedia services," *2005 1st International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM'05)*, 2005.
- [126] P. Weik, D. Vingarzen, and T. Magedanz, "The FOKUS Open IMS Core - A Global Reference Implementation," *IMS Handbook - Concepts, Technologies and Services - Mohamed Ilyas and Syad Ahson*, pp. 113–132, Nov. 2008.
- [127] Fraunhofer Institute FOKUS, "Open source IMS core," <http://www.openimscore.org/>.
- [128] ETSI, "TS 186 008, IMS network testing (INT) IMS/NGN performance benchmark," *V1.2.2*, Jun. 2013.
- [129] 3GPP, "3GPP TS 29.328 v12.4.0, "IP multimedia (IM) subsystem sh interface; signalling flows and message contents ; release 12" (2014)," (*Release 12*), Mar. 2014.
- [130] Google, "Chrome Web Browser," January 2017.
[Online]. Available: <https://www.google.com/chrome/>
- [131] Callflow Website, "Callflow Sequence Diagram Generator," October 2014.
[Online]. Available: <http://sourceforge.net/projects/callflow/>
- [132] IMS Bench Website, "IMS Bench SIPp," April 2014.
[Online]. Available: http://sipp.sourceforge.net/ims_bench/
- [133] D. Thissen, J. M. E. Carlin, and R. Herpertz, "Evaluating the Performance of an IMS/NGN Deployment," in *Informatik 2009: Im Focus das Leben, Beitrage der 39. Jahrestagung der Gesellschaft fur Informatik e.V. (GI), 28.9.-2.10.2009, Lubeck, Proceedings*, ser. LNI, S. Fischer, E. Maehle, and R. Reischuk, Eds., vol. 154. GI, 2009, pp. 2561–2573.

- [134] D. Vingarzan, “[OpenIMSCore-Users] Time measurements on openIMS.”
[Online]. Available:
<https://sourceforge.net/p/openimscore/mailman/message/32544186/>
- [135] A. Munir and A. Gordon-Ross, “SIP-Based IMS Signaling Analysis for WiMax-3g Interworking Architectures,” *IEEE Transactions on Mobile Computing*, vol. 9, no. 5, pp. 733–750, May 2010.
- [136] P. Li, Z. Shi, G. Yu, L. Li, and H. Huang, “Research on Improving Strategies of SIP Session Setup Delay in IMS,” in *2012 International Conference on Computer Science and Electronics Engineering (ICCSEE)*, vol. 1, Mar. 2012, pp. 498–501.
- [137] R. Mahmood and M. Azad, “SIP messages delay analysis in heterogeneous network,” in *International Conference on Wireless Communication and Sensor Computing, 2010. ICWCSC 2010*, Jan. 2010, pp. 1–5.
- [138] D. Cederman, B. Chatterjee, N. Nguyen, Y. Nikolakopoulos, M. Papatriantafilou, and P. Tsigas, “A Study of the Behavior of Synchronization Methods in Commonly Used Languages and Systems,” in *2013 IEEE 27th International Symposium on Parallel Distributed Processing (IPDPS)*, May 2013, pp. 1309–1320.
- [139] Kamilio Website, “Kamilio SIP Server,” June 2017.
[Online]. Available: <https://www.kamilio.org/w/>
- [140] P. Bellavista, A. Corradi, and L. Foschini, “Enhancing Intra-Domain Scalability of IMS-based Services,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 99, no. 1, p. 1, Nov. 2012.
- [141] T. Pedersen, T. L. Guilly, A. P. Ravn, and A. Skou, “A method for model checking feature interactions,” in *2015 10th International Joint Conference on Software Technologies (ICSOFT)*, vol. 1, Jul. 2015, pp. 1–10.

- [142] M. Z. Rafique, Z. S. Khan, M. K. Khan, and K. Alghatbar, “Securing IP-Multimedia Subsystem (IMS) against Anomalous Message Exploits by Using Machine Learning Algorithms,” in *2011 Eighth International Conference on Information Technology: New Generations*, Apr. 2011, pp. 559–563.

Appendix A:

Example Test Report

An example test report from one of the performance tests conducted against the proposed architecture is included here for completeness. It was automatically generated using the IMS Bench program.

It shows the actual configuration that was used to run the tests, the number of machines involved in the test as well as their resource usage. The number of scenarios executed per second are shown, together with the delay characteristics observed during the test run.

As these reports are fairly lengthy, only the relevant figures were extracted and discussed in Chapter 7.

Test Report (icscf, pcscf, scscf, hss, presence)

Summary

This report shows the result of a benchmark run performed by "IMS Bench SIPP", an implementation of the **IMS/NGN Performance Benchmark** suite, **ETSI TS 186.008**.

The test was started on 28-Dec-2015 21:20, and the total time for the test execution was **0h 36m 50s**. The **Design Objective Capacity (DOC)** has **not been reached**. The DOC is **>= 24** scenarios per second.

The following systems and parameters were used for the test.

Role	Server	IP	Nb Users
SUT 1	icscf	172.30.0.226	
SUT 2	pcscf	172.30.0.227	
SUT 3	scscf	172.30.0.228	
SUT 4	hss	172.30.0.229	
SUT 5	presence	172.30.0.201	
Manager	test-manager	127.0.1.1	
TS1	test-manager	172.30.0.40	24000

Parameter Name	Parameter Value	Parameter Info
RingTime	5000	Ringing Time (ms)
HoldTime	30000	Conversation Time (ms)
RegistrationExpire	1000000	Registration Timeout (ms)
TransientTime	30	Time after the start of a step for which data is ignored (in seconds)

The following table shows the average of the key measurements for each step of the test. Each steps is characterized by the requested load, the effective load, the global IHS (total of all Inadequately handled scenarios for this step divided by number of Session Attempts for this step) the scenario IHS (number of inadequately handled scenarios for this step divided by the number of scenario attempts for this step), the CPU utilization and the available Memory on the SUT. The available Memory is expressed in MegaBytes, and the requested and effective loads in Scenarios Attempts Per Seconds (SAPS).

Note that the IHS percentages represented in this table are the number of failures for a step divided by the number of scenario attempts for this step, and so is not the average of (IHS per seconds)

	Pre-registration	Step 1	Step 2	Step 3	Step 4	Step 5	Step 6
Requested load	10	5	5	10	15	20	25
Effective Load	10.00	5.08	4.85	10.29	14.72	19.41	24.87
Ratio ims_rereg %	0.00	18.50	20.83	19.20	19.82	20.40	19.32
Ratio ims_dereg %	0.00	41.83	40.74	39.73	39.94	39.49	39.53
Ratio ims_reg %	100.00	39.67	38.43	41.06	40.24	40.11	41.15
CPU icscf	0.36	0.15	0.15	0.24	0.30	0.39	0.44
CPU pcscf	0.12	0.08	0.09	0.11	0.15	0.15	0.20
CPU scscf	0.25	0.13	0.14	0.22	0.29	0.41	0.51
CPU hss	4.54	1.52	1.88	2.64	4.67	5.94	8.53
CPU presence	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Memory icscf	3770.31	3769.69	3769.21	3768.74	3767.69	3766.33	3764.94
Memory pcscf	3771.55	3765.90	3765.88	3765.63	3764.56	3763.20	3761.35
Memory scscf	3747.26	3730.20	3728.63	3723.79	3716.50	3707.82	3695.96
Memory hss	2254.44	2021.19	1997.42	1976.03	1879.67	1684.38	1534.77
Memory presence	3774.52	3774.54	3774.24	3774.28	3774.35	3774.35	3774.35
SIPP CPU test-manager	0.52	0.45	0.43	0.39	0.44	0.46	0.79
SIPP MEM test-manager	530.80	529.39	528.54	527.48	526.41	525.30	523.87
IHS ims_dereg %	0.00	0.00	0.00	0.00	0.00	0.05	0.08
IHS ims_reg %	0.03	0.00	0.00	0.00	0.00	0.14	0.07
global IHS %	0.03	0.00	0.00	0.00	0.00	0.08	0.06

The following chapters show details on different measurement, like delay between two messages, response time or number of messages per seconds. Each measurement can be represented in one of the four following forms.

1. Evolution in function of the time. On such graphs, the raw information is plotted, like number of messages per seconds, or response time of each scenario. This graph is useful in giving for instance a good idea on the distribution of response times, and it's evolution over the time.
2. Evolution (mean) in function of the time. While previous graph gives a good indication, it may sometimes be easier to see the evolution of the mean of the measurement over a second in function of the time.
3. Histogram. This graph shows the histogram of the measurement, so how many times each value of the measurement occurred.
4. Probability. This graph gives the probability of the measurement to be higher than a certain value. This graph can be used to determine percentile for instances.

For some graphs, a cubic Bezier curve is plotted as well.

1 Scenario Attempts Per Second

1.1 Scenario Attempts Per Second (Mean per second)

1.2 Scenario Attempts Per Second Histogram

1.3 Scenario Attempts Per Second Probability

2 SUT CPU %

2.1 SUT CPU % over time

3 SUT Available Memory [MB]

3.1 SUT Available Memory [MB] over time

4 ALL SIPP CPU %

4.1 ALL SIPP CPU % over time

5 ALL SIPP Free Memory [MB]

5.1 ALL SIPP Free Memory [MB] over time

6 Inadequately handled scenario Percentage

6.1 Inadequately handled scenario Percentage over time

7 Scenario retransmissions - all scenarios

7.1 Scenario retransmissions - all scenarios over time

7 Calling

- 7.1 PX_TRT-SES1: Session Setup Time
- 7.2 PX_TRT-SES2: Session Initiation transversal time
- 7.3 PX_TRT-REL1: Delay Between BYE and 200 OK
- 7.4 PX_TRT-SES3: INVITE and re-INVITE cost
- 7.5 ims_uac : Scenario retransmissions
- 7.5.1 Scenario retransmissions(ims_uac scenario) over time
- 7.6 ims_uac : Inadequately handled scenario Percentage
- 7.6.1 Inadequately handled scenario Percentage(ims_uac scenario) over time
- 8 Messaging
 - 8.1 PX_TRT-PMM1: Message Transmission time
 - 8.2 PX_TRT-PMM2: Message Transmission time (error case)
- 9 Registration
 - 9.1 PX_TRT-REG1: Time of the first register transaction
 - 9.1.1 PX_TRT-REG1: Time of the first register transaction(Registration use case) (Mean per second)
 - 9.1.2 PX_TRT-REG1: Time of the first register transaction(Registration use case) Histogram
 - 9.2 PX_TRT-REG2: Time of the second register transaction
 - 9.2.1 PX_TRT-REG2: Time of the second register transaction(Registration use case) (Mean per second)
 - 9.2.2 PX_TRT-REG2: Time of the second register transaction(Registration use case) Histogram
 - 9.3 ims_dereg : PX_TRT-REG1: Time of the first register transaction
 - 9.3.1 PX_TRT-REG1: Time of the first register transaction(ims_dereg scenario) (Mean per second)
 - 9.3.2 PX_TRT-REG1: Time of the first register transaction(ims_dereg scenario) Histogram
 - 9.4 ims_reg : Scenario retransmissions
 - 9.4.1 Scenario retransmissions(ims_reg scenario) over time
 - 9.5 ims_reg : PX_TRT-REG1: Time of the first register transaction
 - 9.5.1 PX_TRT-REG1: Time of the first register transaction(ims_reg scenario) (Mean per second)
 - 9.5.2 PX_TRT-REG1: Time of the first register transaction(ims_reg scenario) Histogram
 - 9.6 ims_rereg : Time of the re-register transaction
 - 9.6.1 Time of the re-register transaction(ims_rereg scenario) (Mean per second)
 - 9.6.2 Time of the re-register transaction(ims_rereg scenario) Histogram

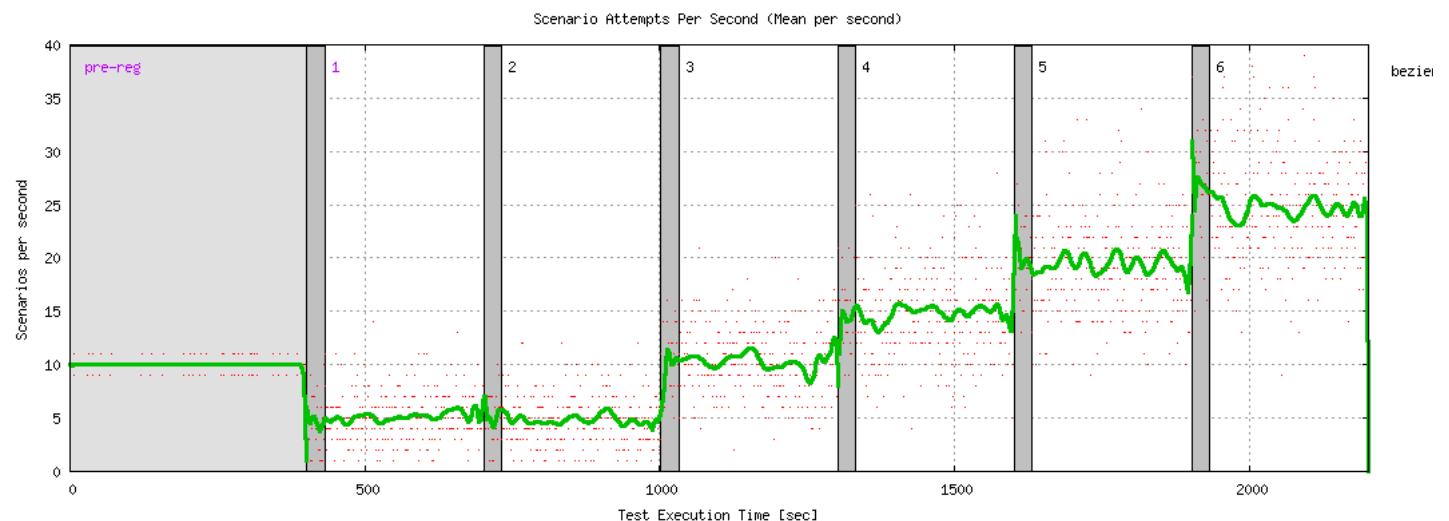
Appendix

1 Scenario Attempts Per Second ↻

This graph represents the number of scenario per seconds generated by the test system. For each step, the generation was based on a Poisson.

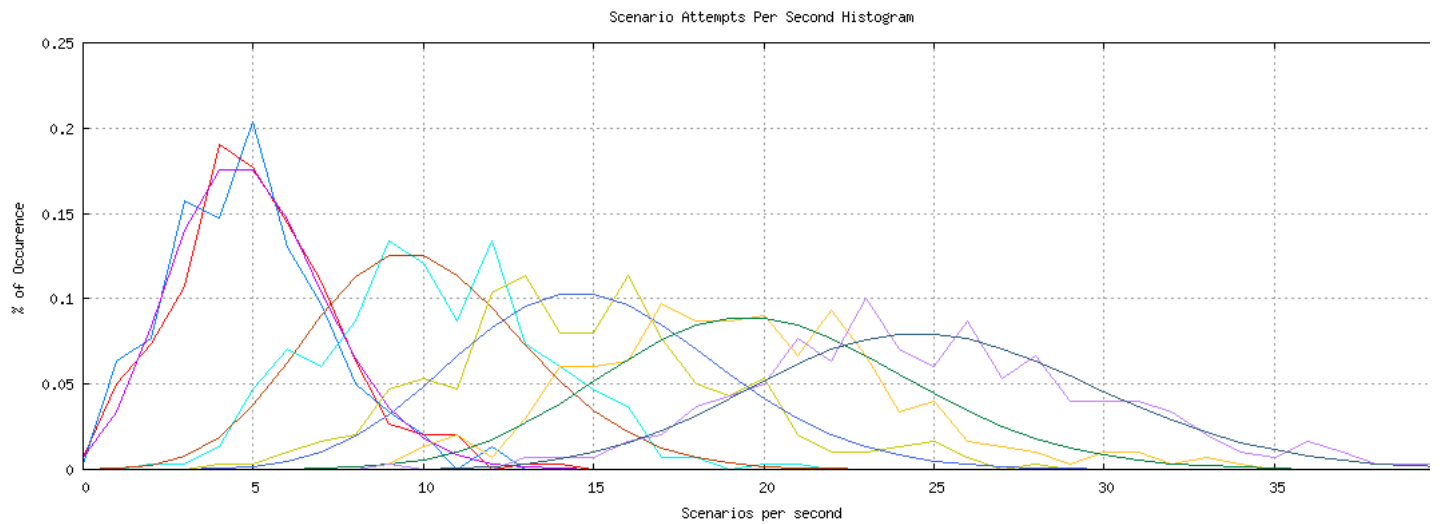
Step	Requested Load	Effective Load								
		Mean	Variance	Standard Deviation	Minimum	Maximum	Percentile 50	Percentile 90	Percentile 95	Percentile 99
Pre-reg	10	10.00	0.14	0.37	9.00	11.00	10.0	10.0	11.0	11.0
1	5	5.08	5.54	2.35	0.00	14.00	5.0	8.0	9.0	11.0
2	5	4.85	5.22	2.28	0.00	12.00	5.0	8.0	9.0	12.0
3	10	10.29	10.38	3.22	2.00	21.00	10.0	15.0	16.0	18.0
4	15	14.72	17.83	4.22	4.00	28.00	14.0	20.0	23.0	26.0
5	20	19.41	20.21	4.50	9.00	34.00	19.0	25.0	27.0	33.0
6	25	24.87	26.66	5.16	9.00	40.00	24.0	32.0	34.0	38.0

1.1 Scenario Attempts Per Second (Mean per second) ↻



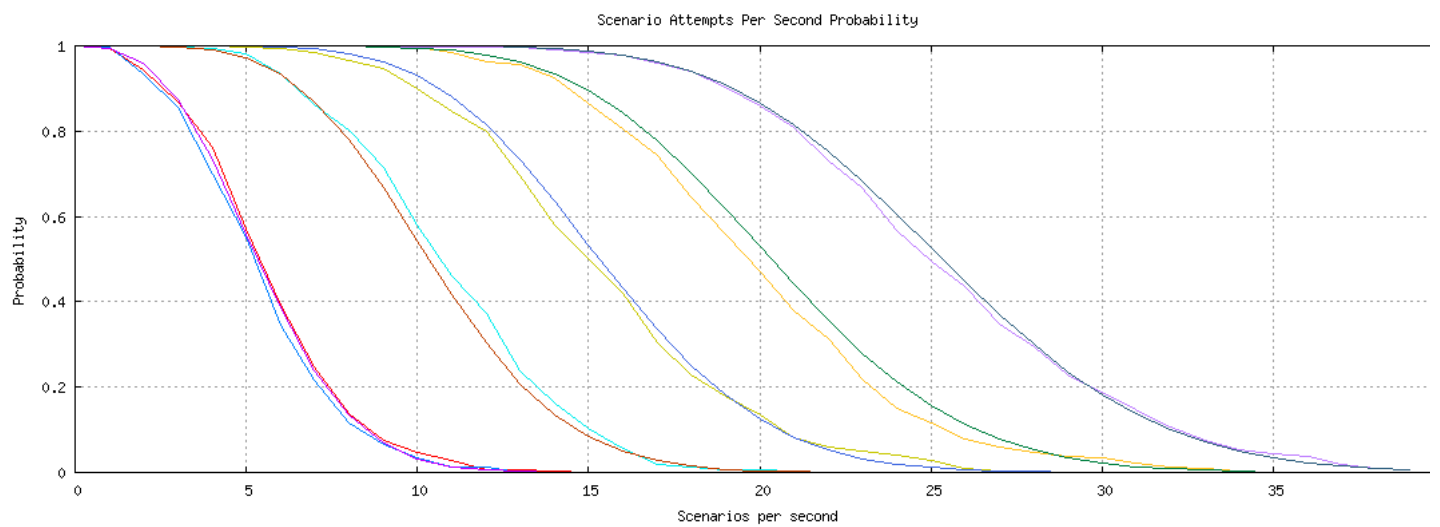
1.2 Scenario Attempts Per Second Histogram ↻

This graph shows the Histogram of the SAPS for each step. It should follow Poisson distributions.



1.3 Scenario Attempts Per Second Probability ↻

This graph shows the probability distribution of the SAPS for each step. It shows the probability that the effective load is higher than x.

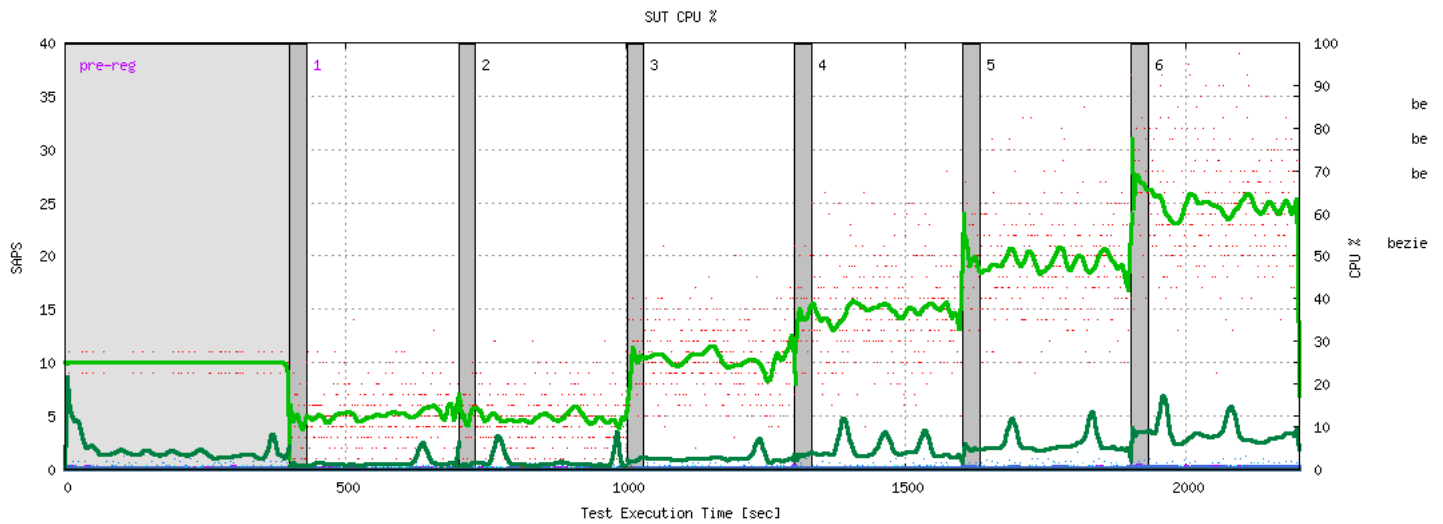


2 SUT CPU % ↻

This graph represents the CPU of the system under test (SUT).

Step	Requested Load	CPU icscf				CPU pcscf				CPU scscf				CPU hss			
		Mean	Standard Deviation	Minimum	Maximum	Mean	Standard Deviation	Minimum	Maximum	Mean	Standard Deviation	Minimum	Maximum	Mean	Standard Deviation	Minimum	Maximum
Pre-reg	10	0.36	0.56	0.00	3.02	0.12	0.25	0.00	1.51	0.25	0.33	0.00	1.50	4.54	6.78	0.00	8.00
1	5	0.15	0.31	0.00	1.52	0.08	0.21	0.00	1.00	0.13	0.24	0.00	1.00	1.52	4.09	0.00	5.00
2	5	0.15	0.34	0.00	2.02	0.09	0.20	0.00	1.00	0.14	0.26	0.00	1.49	1.88	7.43	0.00	9.00
3	10	0.24	0.41	0.00	2.51	0.11	0.25	0.00	1.51	0.22	0.29	0.00	1.02	2.64	4.32	0.00	5.00
4	15	0.30	0.45	0.00	2.53	0.15	0.27	0.00	1.01	0.29	0.36	0.00	1.52	4.67	9.70	0.00	9.00
5	20	0.39	0.60	0.00	4.02	0.15	0.30	0.00	1.52	0.41	0.40	0.00	2.00	5.94	8.92	0.54	9.00
6	25	0.44	0.58	0.00	3.02	0.20	0.35	0.00	1.52	0.51	0.41	0.00	2.00	8.53	10.20	0.56	9.00

2.1 SUT CPU % over time ↻

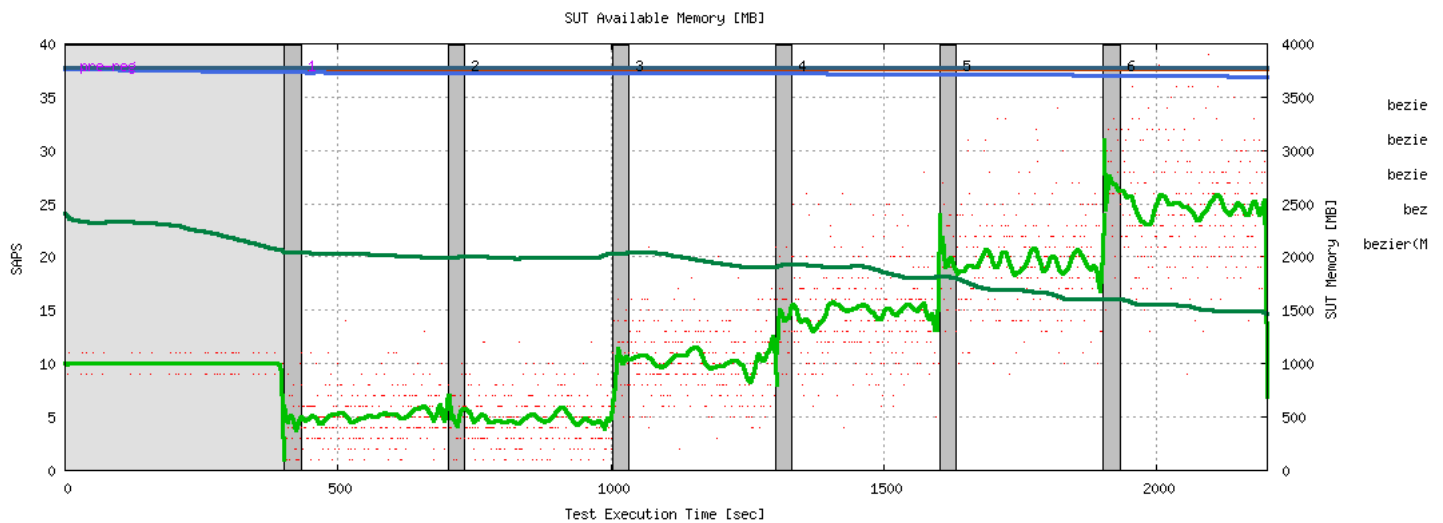


3 SUT Available Memory [MB]

This graph represents the Available memory on the system under test, in MBytes (SUT).

Step	Requested Load	Memory icscf				Memory pcscf				Memory scscf				Memory		Mi
		Mean	Standard Deviation	Minimum	Maximum	Mean	Standard Deviation	Minimum	Maximum	Mean	Standard Deviation	Minimum	Maximum	Mean	Standard Deviation	
Pre-reg	10	3770.31	0.39	3769.60	3773.19	3771.55	3.23	3766.09	3779.79	3747.26	9.48	3730.67	3766.30	2254.44	88.74	20
1	5	3769.69	0.16	3769.26	3770.13	3765.90	0.14	3765.54	3766.16	3730.20	0.23	3729.59	3730.62	2021.19	17.11	19
2	5	3769.21	0.14	3768.92	3769.60	3765.88	0.09	3765.70	3766.11	3728.63	0.58	3727.46	3729.77	1997.42	8.41	19
3	10	3768.74	0.15	3768.28	3769.10	3765.63	0.12	3765.42	3766.10	3723.79	1.47	3721.09	3727.68	1976.03	54.69	19
4	15	3767.69	0.30	3766.86	3768.46	3764.56	0.32	3763.99	3765.60	3716.50	2.30	3712.93	3721.19	1879.67	48.25	18
5	20	3766.33	0.40	3765.57	3767.45	3763.20	0.29	3762.81	3764.24	3707.82	2.65	3703.73	3713.02	1684.38	66.49	15
6	25	3764.94	0.24	3764.32	3765.88	3761.35	0.44	3760.69	3763.01	3695.96	3.60	3689.59	3703.82	1534.77	40.17	14

3.1 SUT Available Memory [MB] over time

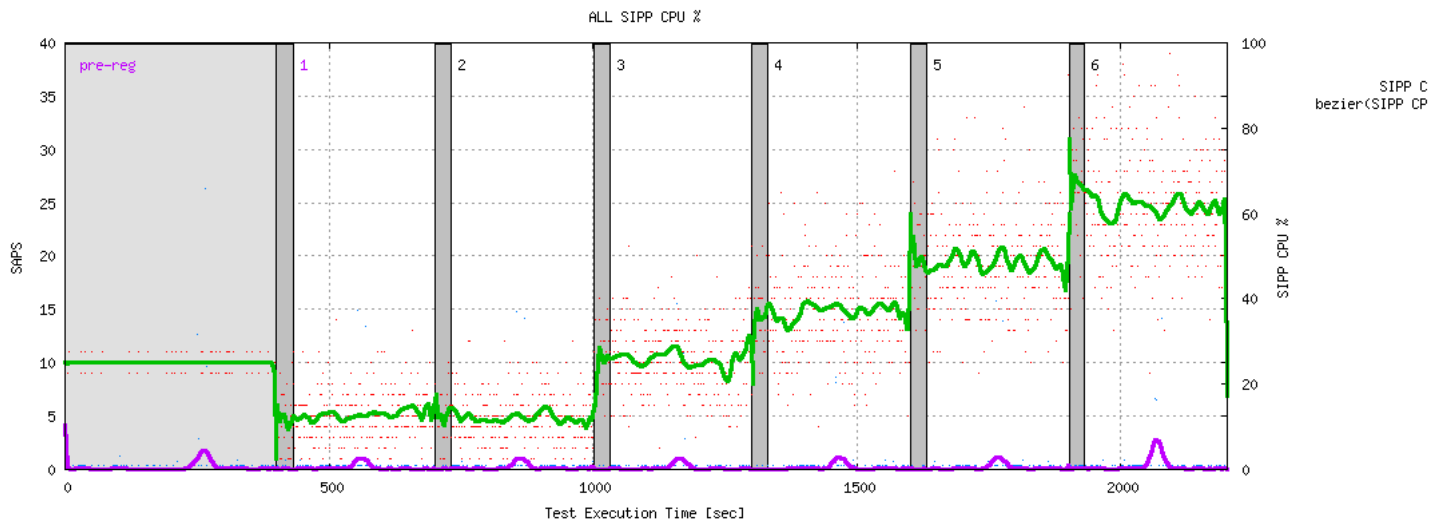


4 ALL SIPP CPU %

This graph represents the CPU of SIPP on ALL Test Machines

SIPP CPU test-manager					
Step	Requested Load	Mean	Standard Deviation	Minimum	Maximum
Pre-reg	10	0.52	3.89	0.00	65.75
1	5	0.45	2.91	0.00	37.37
2	5	0.43	2.97	0.00	37.11
3	10	0.39	2.77	0.00	38.78
4	15	0.44	2.63	0.00	34.34
5	20	0.46	3.04	0.00	39.39
6	25	0.79	6.25	0.00	100.00

4.1 ALL SIPP CPU % over time

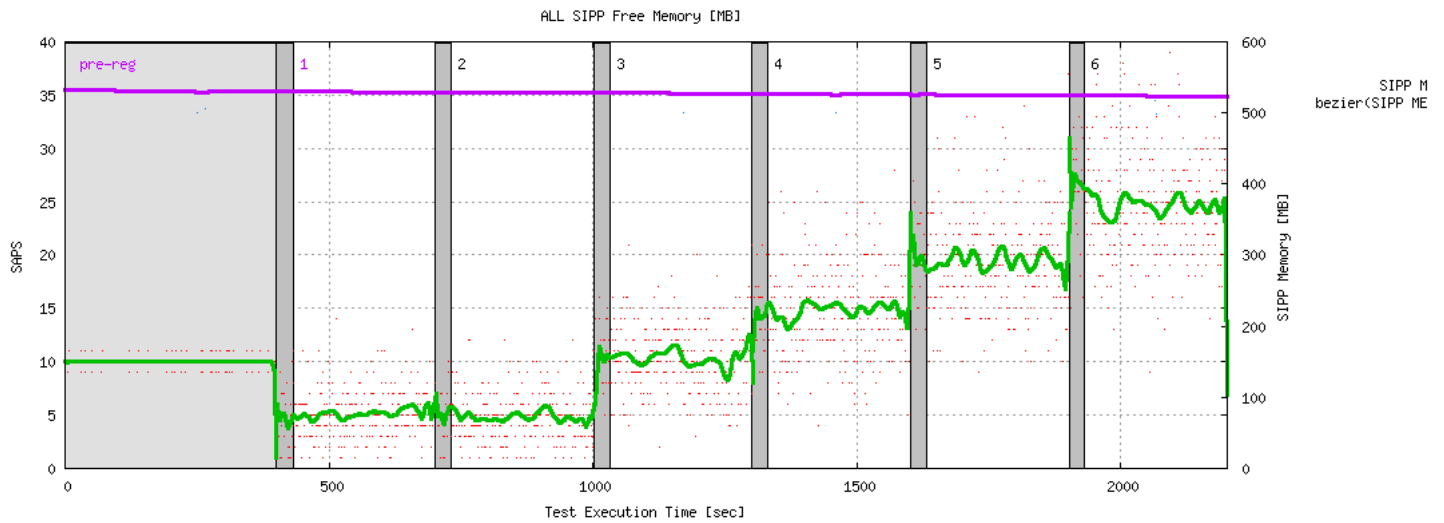


5 ALL SIPP Free Memory [MB] ↻

This graph represents the free memory of SIPP on ALL Test Machines, in MBytes

SIPP MEM test-manager					
Step	Requested Load	Mean	Standard Deviation	Minimum	Maximum
Pre-reg	10	530.80	2.04	501.06	532.42
1	5	529.39	0.23	528.78	529.71
2	5	528.54	0.22	527.85	528.90
3	10	527.48	1.61	500.08	528.01
4	15	526.41	1.53	500.61	526.95
5	20	525.30	0.31	524.41	525.89
6	25	523.87	1.59	497.85	524.60

5.1 ALL SIPP Free Memory [MB] over time ↻



6 Inadequately handled scenario Percentage ↻

This graph represents the percentage of inadequately handled scenarios.

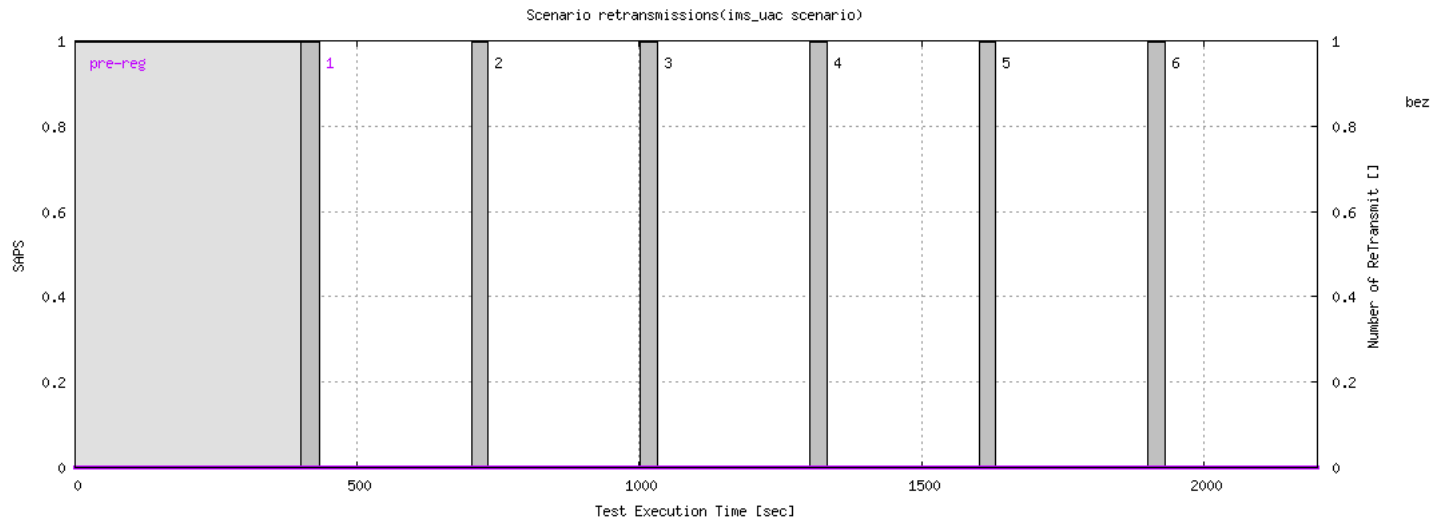
IHS per use_case %					
Step	Requested Load	Mean	Standard Deviation	Minimum	Maximum
Pre-reg	10	0.03	0.50	0.00	10.00
1	5	0.00	0.00	0.00	0.00
2	5	0.00	0.00	0.00	0.00
3	10	0.00	0.00	0.00	0.00
4	15	0.00	0.00	0.00	0.00
5	20	0.09	0.75	0.00	9.52
6	25	0.07	0.59	0.00	6.25

6.1 Inadequately handled scenario Percentage over time ↻

[illegible]

Step	Requested Load	Mean	Standard Deviation	Minimum	Maximum	Percentile 50	Percentile 90	Percentile 95	Percentile 99
Pre-reg	10	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0
1	5	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0
2	5	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0
3	10	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0
4	15	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0
5	20	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0
6	25	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0

7.5.1 Scenario retransmissions(ims_uac scenario) over time ↻

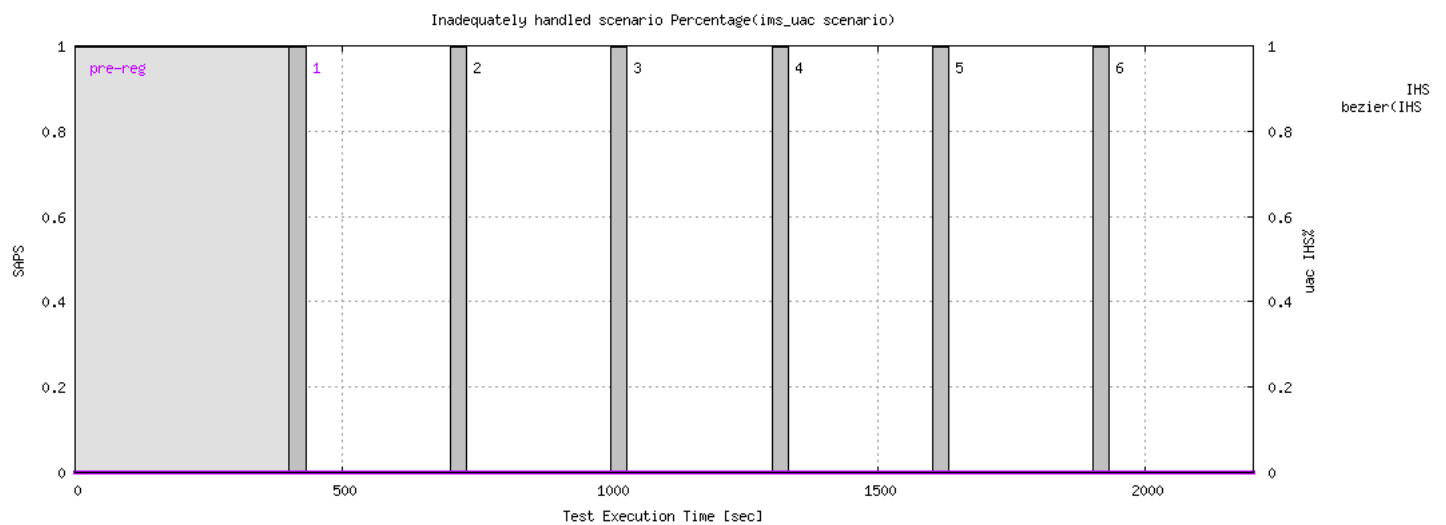


7.6 ims_uac : Inadequately handled scenario Percentage ↻

This graph represents the percentage of Inadequately handled scenarios for the uac.

Step	Requested Load	IHS per scenario %			
		Mean	Standard Deviation	Minimum	Maximum
Pre-reg	10	0.00	0.00	0.00	0.00
1	5	0.00	0.00	0.00	0.00
2	5	0.00	0.00	0.00	0.00
3	10	0.00	0.00	0.00	0.00
4	15	0.00	0.00	0.00	0.00
5	20	0.00	0.00	0.00	0.00
6	25	0.00	0.00	0.00	0.00

7.6.1 Inadequately handled scenario Percentage(ims_uac scenario) over time ↻



8 Messaging ↻

8.1 PX_TRT-PMM1: Message Transmission time ↻

This graph represents the delay between the message and the 200 OK.

8.2 PX_TRT-PMM2: Message Transmission time (error case) ↻

This graph represents the delay between the message and the 404 Not Found.

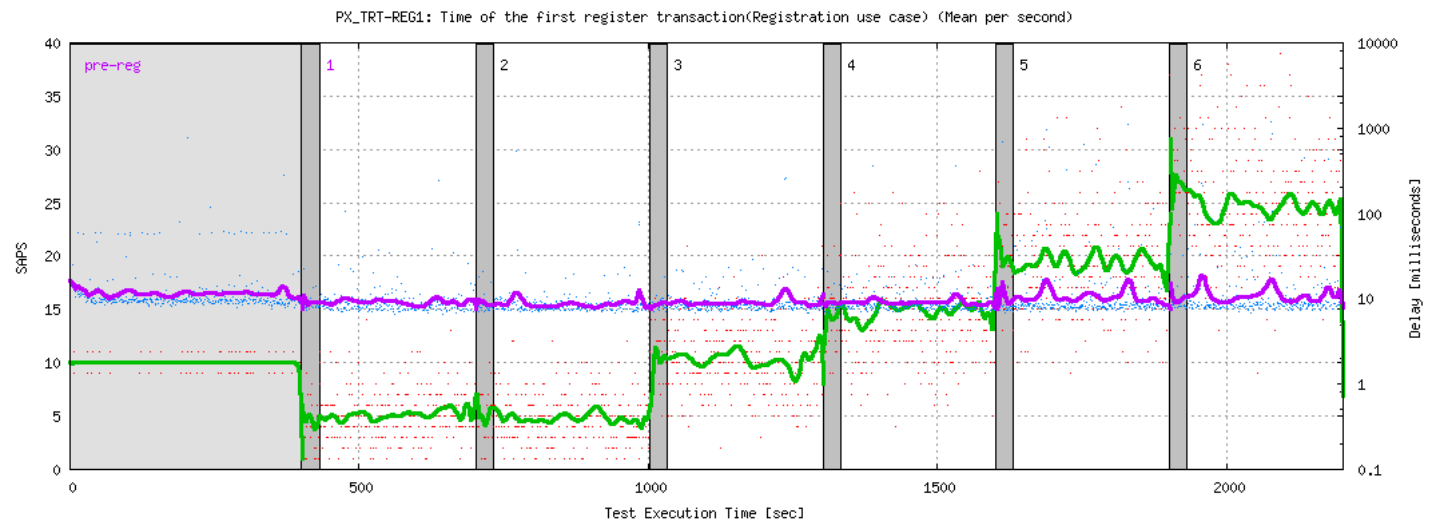
9 Registration ↻

9.1 PX_TRT-REG1: Time of the first register transaction ↻

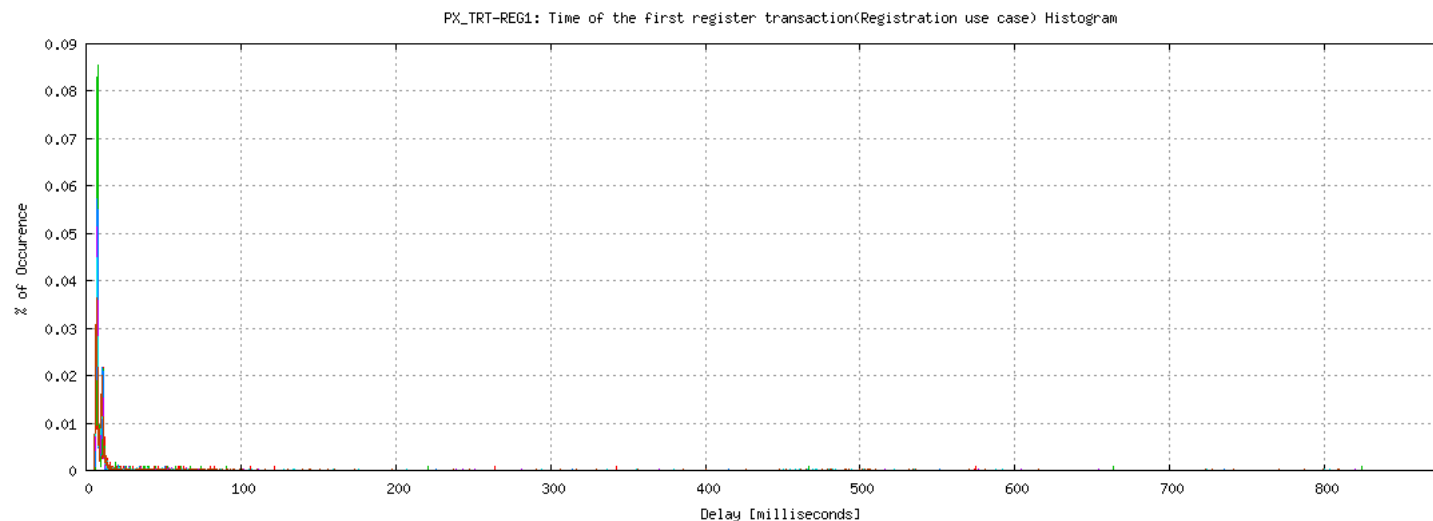
This graph represents the time of the first register transaction in the registration use_cases i.e. the time between the REGISTER and the 401 Unauthorized for all scenarios in the Registration use_case.

		PX_TRT-REG1 (msec)							
Step	Requested Load	Mean	Standard Deviation	Minimum	Maximum	Percentile 50	Percentile 90	Percentile 95	Percentile 99
Pre-reg	10	16.36	126.29	6.38	7500.89	9.0	12.2	14.5	164.8
1	5	10.43	21.35	5.84	574.37	7.7	11.4	12.6	60.0
2	5	11.87	50.58	5.89	1269.59	7.5	11.1	11.7	52.3
3	10	10.40	29.24	5.52	964.92	7.4	11.0	12.4	53.0
4	15	10.74	30.29	5.57	841.51	7.3	11.0	12.8	61.3
5	20	27.84	265.33	5.43	7540.30	7.3	11.3	14.4	494.8
6	25	27.41	215.00	5.47	8743.89	7.6	12.1	17.6	535.6

9.1.1 PX_TRT-REG1: Time of the first register transaction(Registration use case) (Mean per second) ↻



9.1.2 PX_TRT-REG1: Time of the first register transaction(Registration use case) Histogram ↻

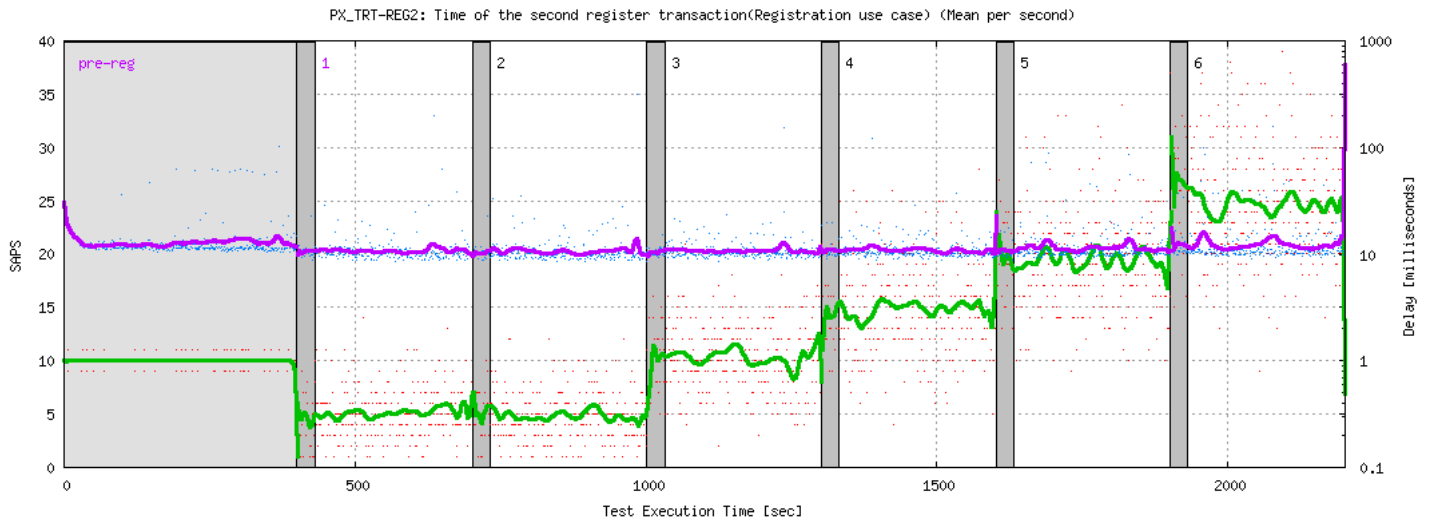


9.2 PX_TRT-REG2: Time of the second register transaction ↻

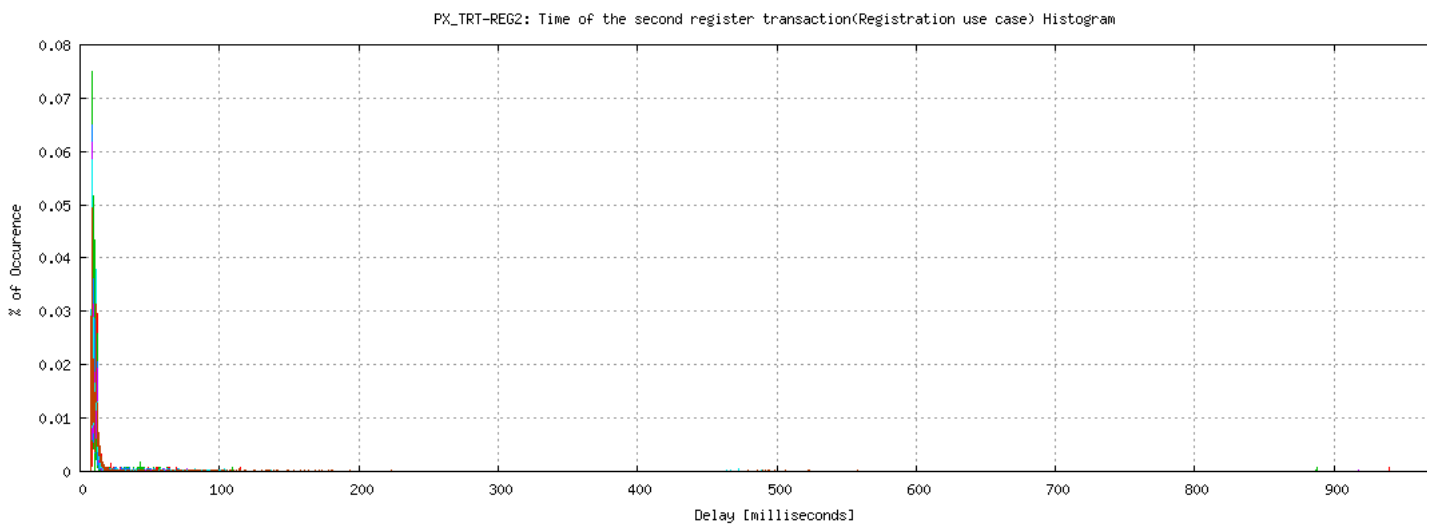
This graph represents the time of the second register transaction in the registration use_cases, i.e. the delay between the second REGISTER and the 200 OK.

		PX_TRT-REG2 (msec)							
Step	Requested Load	Mean	Standard Deviation	Minimum	Maximum	Percentile 50	Percentile 90	Percentile 95	Percentile 99
Pre-reg	10	14.03	29.33	9.64	783.54	11.7	13.8	17.3	38.8
1	5	11.67	26.91	8.26	939.64	10.0	12.4	13.4	33.9
2	5	11.64	26.41	8.16	887.43	9.8	12.2	13.3	43.9
3	10	11.21	20.03	8.06	980.93	9.8	12.2	14.1	35.6
4	15	11.35	16.57	7.76	917.53	9.8	12.7	14.4	49.3
5	20	12.51	20.63	7.82	498.63	9.9	13.2	15.4	76.9
6	25	14.22	35.11	7.73	1758.11	10.3	14.2	17.3	110.5

9.2.1 PX_TRT-REG2: Time of the second register transaction(Registration use case) (Mean per second) ↻



9.2.2 PX_TRT-REG2: Time of the second register transaction(Registration use case) Histogram

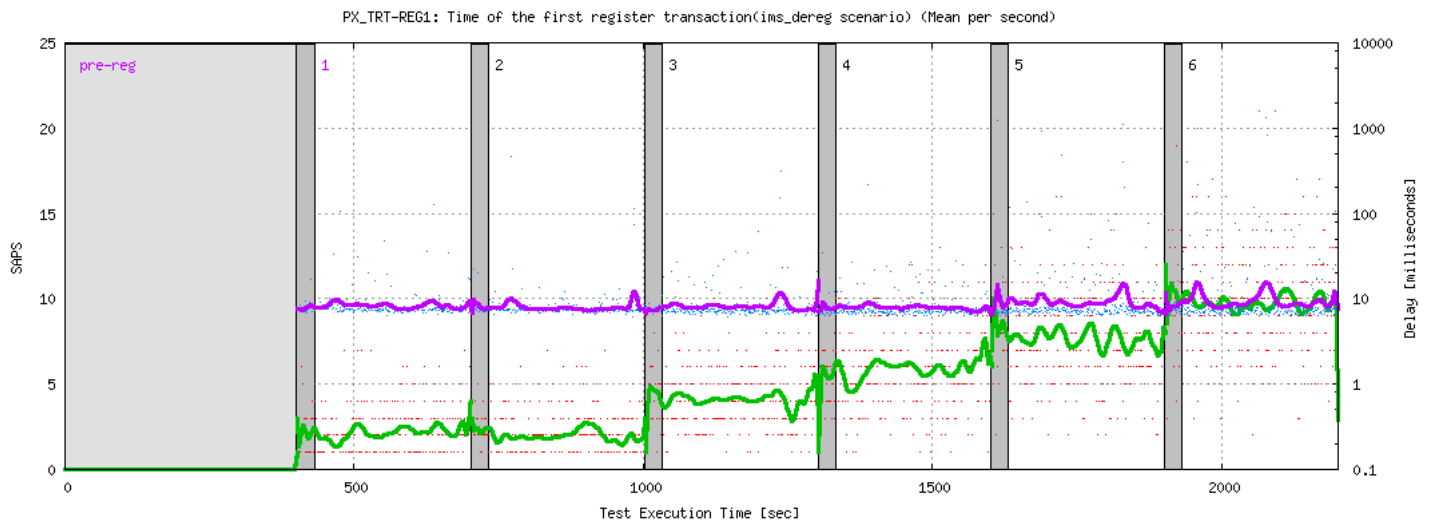


9.3 ims_dereg : PX_TRT-REG1: Time of the first register transaction

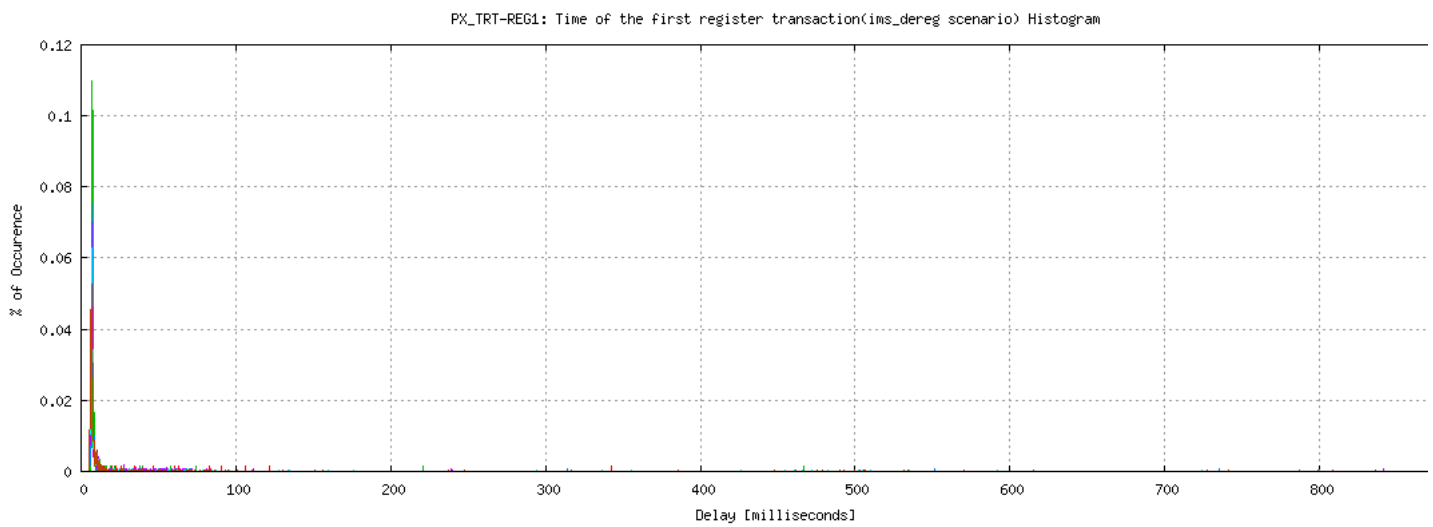
This graph represents the time of the first register transaction i.e. the time between the REGISTER and the 401 Unauthorized.

		PX_TRT-REG1 (msec)							
Step	Requested Load	Mean	Standard Deviation	Minimum	Maximum	Percentile 50	Percentile 90	Percentile 95	Percentile 99
1	5	9.46	15.88	5.84	342.45	7.5	9.1	10.9	60.8
2	5	9.06	21.05	5.89	467.09	7.4	8.5	9.0	38.2
3	10	9.61	27.97	5.52	735.43	7.2	8.7	11.1	53.0
4	15	8.87	21.82	5.57	841.51	7.0	8.8	11.0	53.2
5	20	27.31	246.00	5.43	7487.09	6.9	10.0	14.1	504.9
6	25	24.93	199.40	5.47	8743.89	6.9	10.8	18.4	533.8

9.3.1 PX_TRT-REG1: Time of the first register transaction(ims_dereg scenario) (Mean per second)



9.3.2 PX_TRT-REG1: Time of the first register transaction(ims_dereg scenario) Histogram

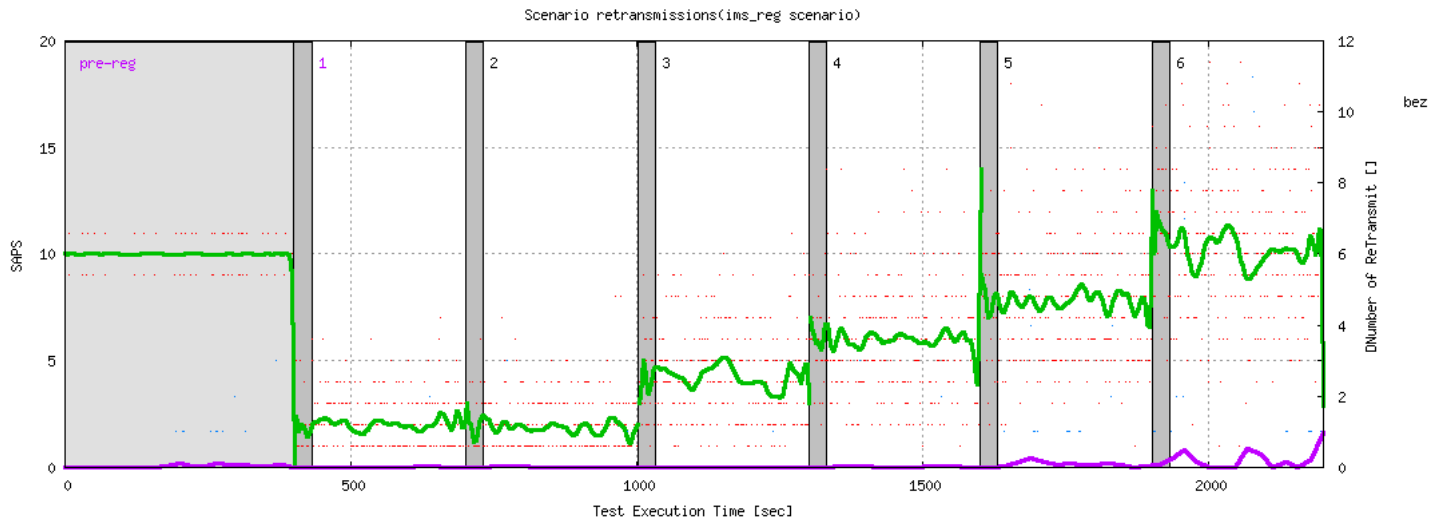


9.4 ims_reg : Scenario retransmissions

This graph represents the number of retransmissions per seconds for this scenario.

Step	Requested Load	RETRANSMIT							
		Mean	Standard Deviation	Minimum	Maximum	Percentile 50	Percentile 90	Percentile 95	Percentile 99
Pre-reg	10	0.04	0.23	0.00	3.00	0.0	0.0	0.0	1.0
1	5	0.01	0.12	0.00	2.00	0.0	0.0	0.0	0.0
2	5	0.01	0.18	0.00	3.00	0.0	0.0	0.0	0.0
3	10	0.00	0.06	0.00	1.00	0.0	0.0	0.0	0.0
4	15	0.02	0.21	0.00	3.00	0.0	0.0	0.0	0.0
5	20	0.09	0.48	0.00	5.00	0.0	0.0	1.0	4.0
6	25	0.17	1.07	0.00	11.00	0.0	0.0	1.0	8.0

9.4.1 Scenario retransmissions(ims_reg scenario) over time

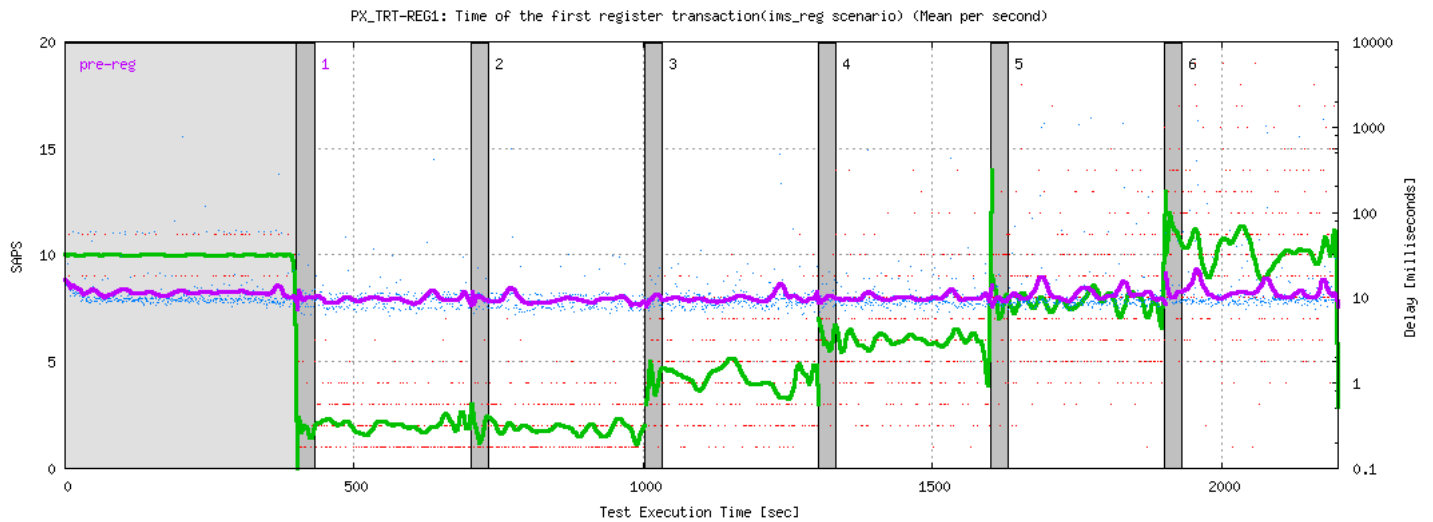


9.5 ims_reg : PX_TRT-REG1: Time of the first register transaction ↻

This graph represents the time of the first register transaction i.e. the time between the REGISTER and the 401 Unauthorized.

		PX_TRT-REG1 (msec)							
Step	Requested Load	Mean	Standard Deviation	Minimum	Maximum	Percentile 50	Percentile 90	Percentile 95	Percentile 99
Pre-reg	10	16.36	126.29	6.38	7500.89	9.0	12.2	14.5	164.8
1	5	11.47	25.95	6.04	574.37	8.7	11.7	13.4	59.6
2	5	14.90	69.49	5.89	1269.59	8.1	11.4	12.9	61.4
3	10	11.17	30.42	5.83	964.92	9.5	11.4	13.1	52.3
4	15	12.57	36.68	5.58	820.03	9.6	11.6	13.8	70.6
5	20	28.37	283.20	5.65	7540.30	9.6	11.9	14.5	471.4
6	25	29.79	228.91	5.63	7507.56	9.8	12.8	17.2	535.9

9.5.1 PX_TRT-REG1: Time of the first register transaction(ims_reg scenario) (Mean per second) ↻



9.5.2 PX_TRT-REG1: Time of the first register transaction(ims_reg scenario) Histogram ↻

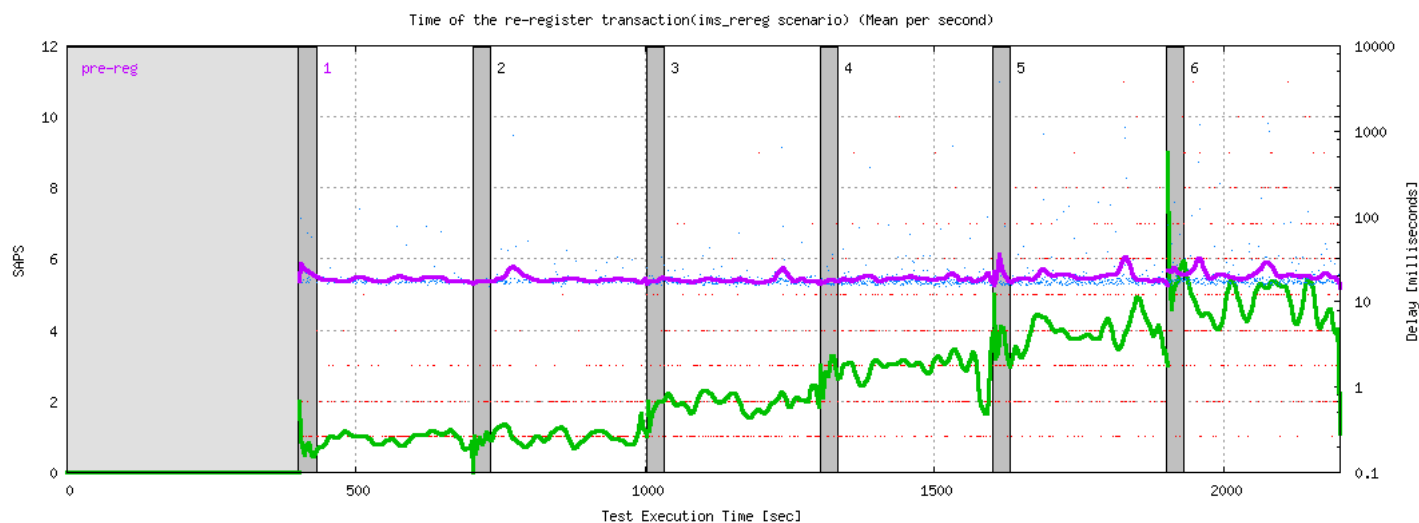


9.6 ims_rereg : Time of the re-register transaction ↻

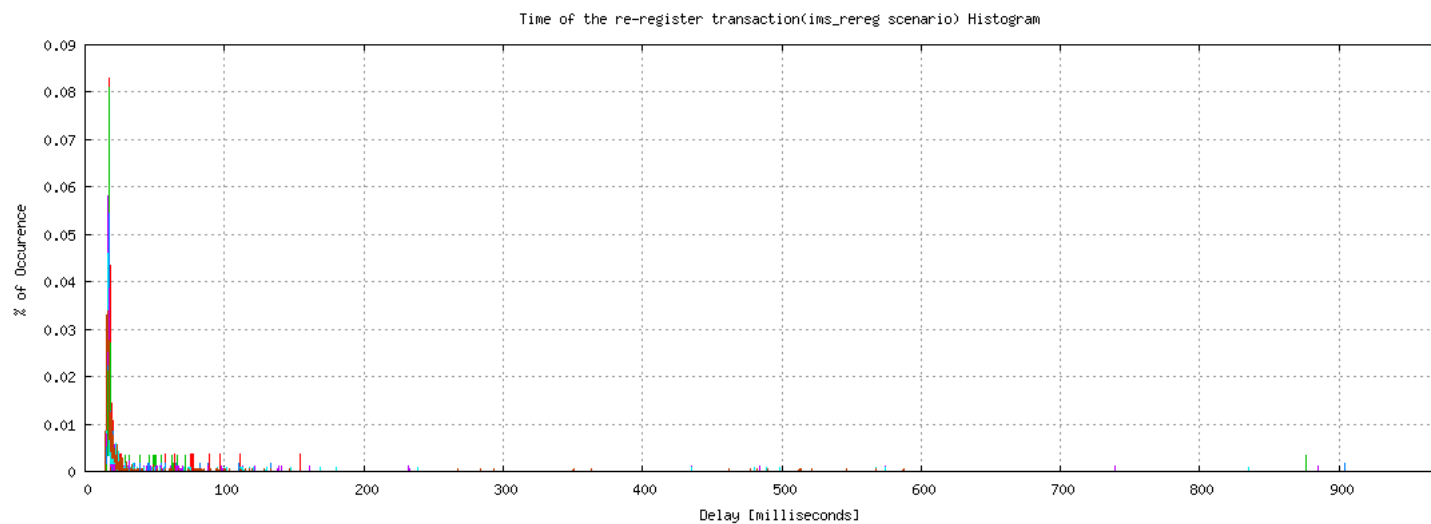
This graph represents the time of re-register transaction i.e. the time between the REGISTER and the 200 OK.

PX_TRT-REG4 (msec)									
Step	Requested Load	Mean	Standard Deviation	Minimum	Maximum	Percentile 50	Percentile 90	Percentile 95	Percentile 99
1	5	19.87	13.73	14.62	154.37	17.3	19.2	21.9	96.7
2	5	21.53	50.21	15.02	876.07	17.2	19.4	22.7	65.9
3	10	21.55	54.62	14.62	985.90	16.8	20.0	23.1	82.5
4	15	25.18	70.74	14.52	1094.21	16.6	21.1	27.6	161.2
5	20	35.41	241.30	14.13	7555.98	16.6	22.3	27.1	489.0
6	25	31.37	113.44	14.38	1872.92	16.8	23.7	33.7	482.4

9.6.1 Time of the re-register transaction(ims_rereg scenario) (Mean per second) ↻



9.6.2 Time of the re-register transaction(ims_rereg scenario) Histogram ↻



Appendix ↻

The following information is also available for the test

Parameter Name	Parameter Value	Parameter Info
rand_seed	1451337627	Value used to initialize the random number generators
prep_offset	2000	Time (ms) for scenario preparation (user reservation, etc.) prior to actual execution
highest_measured_time_offset	41	Highest time offset observed at startup between any test system and the manager (microseconds)

System	Command Line
TS1	./sipp -id 1 -i 172.30.0.40 -user_inf ./ims_users_1.inf -rmctrl 172.30.0.40:5000 172.30.0.101:4060 -trace_err -trace_cpumem -trace_scen -trace_retrans
Manager	../manager -f manager.xml
SUT 1	./cpum 172.30.0.40
SUT 2	./cpum 172.30.0.40
SUT 3	./cpum 172.30.0.40
SUT 4	./cpum 172.30.0.40
SUT 5	./cpum 172.30.0.40