IP Multimedia for Municipalities

The supporting architecture

DAN PETERSTRÖM

KTH Information and Communication Technology

Master of Science Thesis
Stockholm, Sweden 2009

TRITA-ICT-EX-2009:103

IP Multimedia for Municipalities:
The supporting architecture

Dan Peterström
danpeter@kth.se

Aug 18, 2009

Examiner and academic advisor: Prof. G. Q. Maguire Jr., KTH
Industrial advisor: Per Ljungberg, Ericsson

Abstract

Fiber deployment is becoming popular and is seen as a way to increase a community's attractiveness to new inhabitants and companies. A new Open Access network is emerging, leading to a more horizontal network architecture. Combining this architecture with IMS enables developers to easily develop new and attractive services. To facilitate the development of new IMS services there needs to be an easy to use development environment and a reliable hardware/software platform upon which to deploy them.

This thesis project will explore the design, development, and evaluation of new IMS applications targeted at municipal networks as well as the service platforms they are deployed on. The thesis will also examined what role IMS plays in Municipalities and why they may need a tailored IMS solution.

During the thesis a reference network for Municipalities was put together and tested. Different service platforms were tested and evaluated.

Sammanfattning

Fiberutbyggnad har blivit mycket populärt och ses som ett sätt att öka en stads attraktivitet för nya invånare och företag. Ett nytt öppen nät håller på att växa fram, detta leder till en mer horisontell nätverksarkitektur. Kombinera denna arkitektur med IMS så kan utvecklare lättare utveckla nya och attraktiva tjänster. För att underlätta utvecklingen av nya IMS-tjänster måste det finnas ett lättanvänd utvecklingsmiljö och en pålitlig hårdvaru/mjukvaru plattform att installera dem på.

Detta examensarbete kommer att undersöka design, utveckling och utvärdering av nya IMS-lösningar riktade till standsnät samt de tjänsteplattformar de används på. Rapporten kommer även granska vilken roll IMS spelar i stadsnät och varför de kan behöva en skräddarsydd IMS-lösning.

Under examensarbetet byggdes och testades ett referensnätverk. Olika tjänsteplatformar testades och utvärderades.

Acknowledgements

I would foremost like to thank Professor Gerald Q. Maguire Jr. for his support during this thesis. He is very knowledgeable and has provided a lot of support and insight.

I would also like to thank my industrial advisor Per Ljungberg for giving me the opportunity to do my thesis at Ericsson and providing insight into Municipalities and open access.

Special thanks to Ruth Pallares and Javier Arenalez Castrodeza at Ericsson in Madrid for their help during my stay. Tanks also to Friedrich Eltester, Monica Madrid, Dejan Lugonja, Mats Peterström and Anders Orrevad for support and feedback.

Table of Contents

List of Figures

List of Acronyms and Abbreviations

3GPP	3rd Generation Partnership Project
API	Application Programming Interface
AS	Application Server
B2BUA	Back to Back User Agent
BGCF	Breaking Gateway Control Function
CCTV	Closed-circuit television
COTS	Commercial Off The Shelf
EJB	Enterprise Java Beans
ESSIM	Ericsson TSP SAF Simulator
FTTH	Fiber To The Home
HLR	Home Location Register
HSS	Home Subscriber Server
IETF	Internet Engineering Task Force
IITB	IMS In The Box
IMS	IP based Multimedia Subsystem
ISP	Internet Service Provider
LDAP	Lightweight Directory Access Protocol
LOTC	Linux Open Telecom Cluster
M2M	Machine to Machine
MMAS	Multimedia Application Server
MRF	Media Resource Function
NBI	North Bound Interface
NGN	Next Generation Network
OAM	Operation and Maintenance

OMP	Open Multimedia Platform
P2P	Peer to Peer
PGM	Precense and Group Management
PSTN	Public Switches Telephone Network
PoC	Push to talk over Cellular
RAID	Redundant Array of Inexpensive Disks
URI	Unirform Resource Identifier
SAF	Service Availeability Forum
SDS	Service Development Studio
SGCS	Sun Glassfish Communication Server
SIP	Session Initiation Protocol
SNMP	Simple Network Management Protocol
TISPAN	Telecommunications and Internet converged Services and Protocols for Advanced Networking
TSP	Telecom Service Platform
XCAP	XML Configuration Access Protocol
XDMS	XML Document Management Server
VPN	Virtual Private Network
WDM	Wavelength-Division Multiplexing

1 Introduction

1.1 Municipal Networks

In most cities today there is a cable network (coaxial) for providing cable TV services, a twisted pair copper access network for analog and/or digital telephony and fiber and/or cellular network for voice and Internet services. Traditionally, each of these different kinds of networks was implemented as a vertical network, each with its own transport network, routers, and services. This can be very inefficient as each network operator has to invest in their own infrastructure. These vertical silos also make it hard for new players to enter the market, as they too would need to invest in their own network. Compared to the two legacy alternatives (copper and coaxial cable), fiber is the obvious choice for fixed networks for the foreseeable future. If a community/municipality invests in a fiber network as a municipal resource, they can create a horizontal network architecture that anyone can use. This type of fiber deployment is becoming very popular in smaller communities as a way to increase a community's attractiveness to new inhabitants and companies, hence the term municipal network. This approach is also attractive for (housing) cooperatives that can offer a fiber network to their members, who no longer need to pay for separate networks and at the same time a modern fiber network, offers much greater bandwidth and a brighter future path for evolution.

Fiber is relatively inexpensive to produce, as it is either plastic or silica (glass). However, the cost when deploying fiber is mostly the cost of installation, especially the cost of putting fiber into the ground -- if digging is necessary. One of the reasons for the success of Stockholm's STOKAB [4] is that when someone is already digging up a street or sidewalk, the incremental cost of installing fiber (or conduits through which fiber or other cables can be pulled later) is very low.

The bandwidth of fiber is very large, the limiting factor for high throughput data communication is generally the rate at which the lasers in the endpoints can modulate a signal. With technologies such as wavelength-division multiplexing (WDM) and dense wavelength division multiplexing (DWDM) it is possible to put many optical signals onto the same fiber. Instead of transmitting data with one wavelength in each fiber, WDM sends multiple different wavelengths in the same fiber, multiplying the aggregate bandwidth. This technology is beneficial for older fiber networks that have reached their single optical channel capacity, as WDM increases the effective capacity – avoiding the need to install more fiber.

Mobile broadband is becoming popular as a content delivery network due to increasing bandwidths and decreasing price. However, in a municipal setting wireless is often view as only a complement to the wired network. The reason for this has to do with competition from national and regional wireless operators. The municipality cannot block a national operator from operating in their municipality and there is also a policy problem with having government subsidized wireless networks competing with privately owned networks. However, there are some exceptions to this and there are communities [1] who have used wireless local area network (WLAN) extensions of their municipal network to provide public (often free or low cost) Internet access.

A municipal network can be viewed as three layers. Each of these layers can be operated and owned by different operators. These three layers define a horizontal network architecture and consist of: Service providers, Communication operators, and a passive network operator. (See Figure 1)

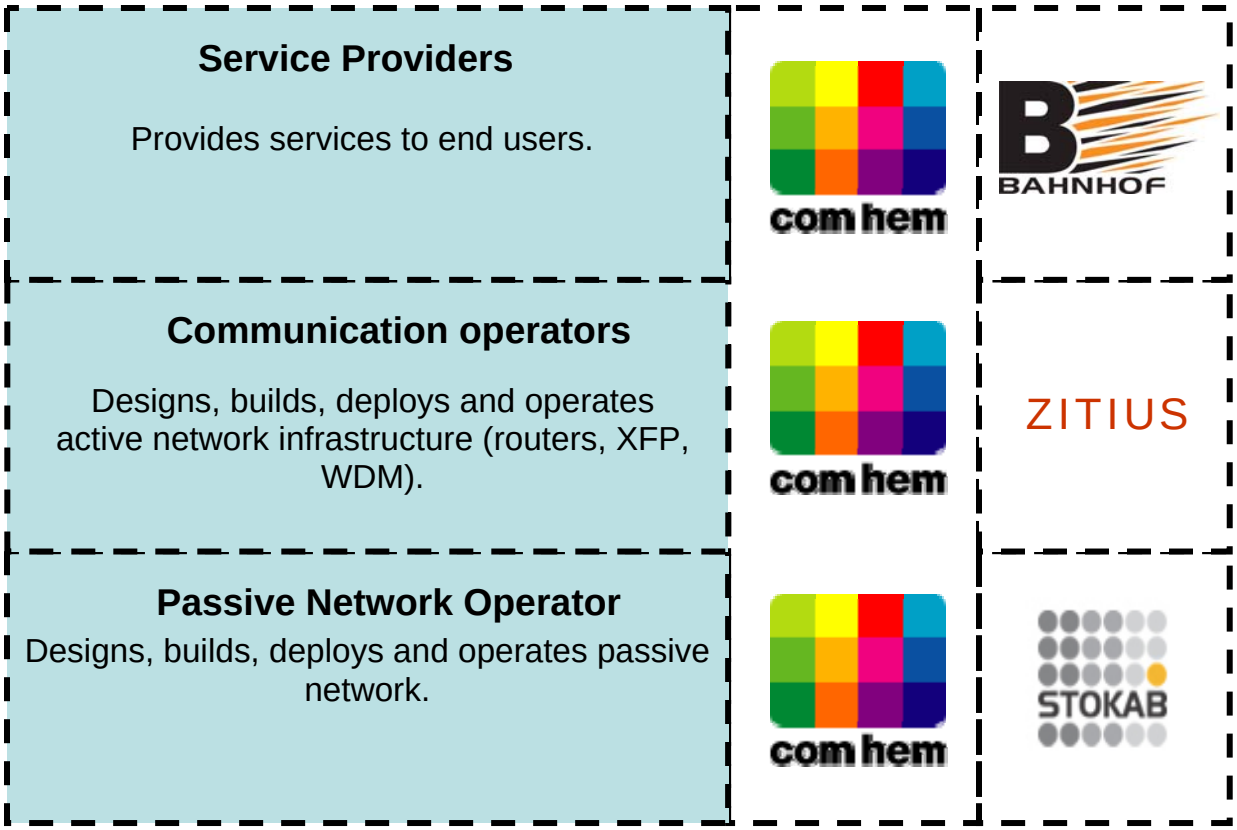

Figure 1: Vertical- vs. Horizontal network

Service Providers

A service provider is anyone providing a service. Such a service provider might be a general purpose Internet service provider (ISP). This service could be IPTV, voice over IP (VoIP), or any other Internet based service. The service provider could be a communication operator or a 3rd party provider. Note that a third party provider might *not* be connected to the same passive network operator as the other parties or party; i.e., they might have their own communication operator and utilize another physical network – however, they have an IP address (or addresses) and are reachable by customers of their service. For example see Banhof [2].

Communication Operators

The communication operator is responsible for the active network infrastructure, specifically the routers, switches, and provisioning. This type of operator might be a "bitpipe" provider, i.e. they do not provide any services other that connectivity between end-points or between end-points and a network interchange point. For example see Zitius [3].

Passive network operator

A passive network operator is responsible for the construction and maintenance of the fiber/wireless broadband network. For example, such an operator might provide dark fiber to be used for services by communication operators. The owner of the network is typically the city itself, as in the case of STOKAB [4] in Stockholm

The benefit of adopting a horizontal network architecture is that it enables an open access strategy. This open access strategy means that fiber deployment is part of city planning and development, rather than each operator having to deploy their own network and dig up the streets. This last aspect is another added benefit that the municipality gains, as the risk to other buried infrastructure is decreased – since the street/sidewalk only needs to be dug up once, rather than once per network operator. This further illustrates the advantages of installing conduits when any project requires digging up a street/sidewalk.

The installation of dark fiber creates a passive network (provided by the passive network operator) that anyone can use it with the same terms and conditions - providing equal footing and fostering competition. By sharing the physical network a lower capital investment is needed to be to offer services. Avoiding this large capital investment can translate into higher profitability for the operator, an earlier time to profit for each of the operators, and/or lower end-user prices. The passive network model eliminates the vertical network model and with it profitable franchise fees. Operators will no longer pay exorbitant fees to be the only operator in a municipality network.

Good references to planned municipalities and the smart city concept are the cities of Johannesburg [5] in South Africa and King Abdullah City [6] in Saudi Arabia.

1.2 Municipal Services

A municipal network differs from a large network operator's network in several ways. A municipality typically has a more homogenous user group, because the network covers a smaller area. It is easier to provide some services as a package to a smaller community, than to offer a package to a country wide network that has users with different requirements. For example, it would be easy to deliver an IPTV service to a municipality network, if the IPTV service provider knows that all residents have Fiber to the Home (FTTH), hence they have the necessary bandwidth for the service. In comparison, offering this to customers who may have a wide variety of different access networks – with widely varying link throughputs is much harder from both a technical and business perspective. IP telephony service providers would know that the all-IP network core enables VoIP using SIP (section 2.3.2). If the operator also provides an IMS (section 2.3) infrastructure, then the IP telephony provider could utilize the authentication, provisioning, and charging offered by IMS. The government can also benefit from having a high throughput and high capacity network, for example, to deploy CCTV based surveillance systems, blue light systems (i.e., emergency services), and remote health monitoring. This advanced service oriented municipal network can be the basis for a *smart city*.

1.2.1 Added value to a municipality by sector:

Some of the most important sectors that may gain from the presence of municipal network are:

- **Business** – competitive pricing of high throughput and high capacity networking by several operators can lead to lower prices, an open network can facilitate a business selecting best of breed offers for services, distributed sensors can enable businesses to exploit knowledge of the microclimate – thus optimizing their operations. High bandwidth virtual private networks (VPNs) lead to more efficient remote backup and can link a branch office to the home office. High definition video conferencing can be used between offices and business partners.
- **Public Safety** – public safety officials can have high bandwidth access to surveillance cameras in real-time. A common command and control center can gather information from sensors and the public (112), then use the network to assign missions (with positioning) to the appropriate blue light units.
- **Education** – remote e-learning can reduce the need for transportation of pupils and teachers; high bandwidth connectivity can allow participation in remote areas to participate from home. E-libraries and Internet sources (such as Wikipedia) gives access to a wide array of educational materials.
- **Transportation** – public transportation can have expedited operations by interacting with the traffic sensing and traffic light system. Real-time traffic congestion monitoring can be used to implement dynamic speed limits. Public transportation and traffic information can be displayed via wireless networks in real time at bus stops.
- **Healthcare** – public health services can provide greater services to their customers at home, reducing the amount of hospitalization necessary. Doctors can reduce their travel time and consult with patients about tests results and show images via videoconference.
- **Utilities** – utilities can have better access to meters to better match production with consumption, with remote control they can even moderate consumption allowing some appliances to be operated when demand is low – rather than simply based on a timer; increased metering can reveal where there are unexpected losses (for example due to water/gas/steam/… leaks).

1.2.2 User services:

Some of the user services that may gain from the presence of municipal network are:

- **Triple play**: providing Voice, Video and TV services via the same network.
- **Internet**: High speed reliable Internet connection with several IP addresses per subscriber.
- **Data Services**: Remote backup for companies and individuals. A home gateway for home users that can control many devices in the home. Media servers can store or cache media.
- **Wireless Service**: Offering connectivity and mobility, while maintaining the same user identity independent of device.
- **Security**: Managed security with firewalls. Deploying water/fire/gas/… detectors to save both property and human lives.
- **Enterprise and VPN Solutions**: Work from home and connect to a remote office. Managed VoIP telephony system offering the same identity independent of device.
- **Education and Healthcare**: Attend lectures from home. Browse large libraries from home. Consult their doctor about non-critical issues from home without the need to travel to the hospital.
- **Entertainment and user created content**: While visiting a friend share your vacation pictures via your home gateway/media server. Watch high definition IPTV and TV on demand via a network attached media server (at home or provided by a service provider).

2 Background

2.1 Network Model

As described in section 1.1, a municipal network can be structured into layers. The current Ericsson network model for this layering is shown in Figure 2.

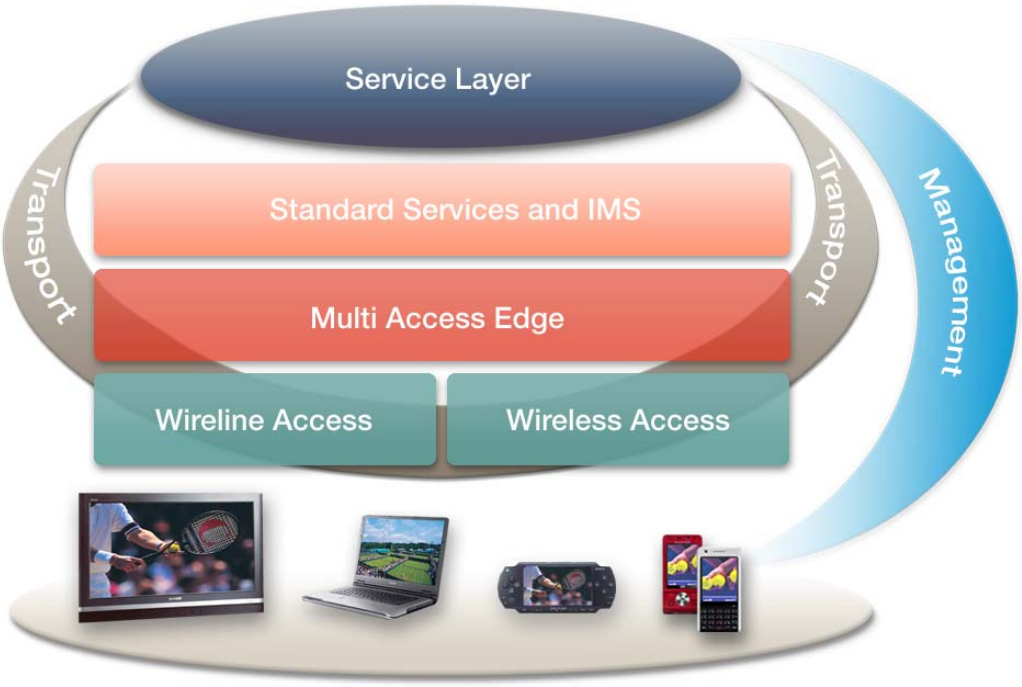

Figure 2: Ericsson Network Model, Copyright Ericsson AB (used with permission)

1. At the bottom of a municipal network is the physical transport layer. It consists of a high speed fiber backbone reaching all homes and businesses (i.e., we assume that it implements fiber to the home - FTTH). Wireless hotspots extend this backbone with high speed wireless connectivity when users are not in (or near) their home and moving around in the community. This wireless network could be a combination of WLAN, WIMAX, and 3G/LTE.
2. On top of the transport layer is the Multi Access Edge layer. This layer consists of routers and gateways that provide seamless switching between the different network technologies creating a homogenous network. In today's networks, this will be an IP network layer, i.e., all of the traffic will be carried using IP based protocols – thus this is really an internetworking layer. To enable large numbers of devices to be connected to this device, this layer probably implements IPv6.
3. The next layer contains the IMS core. (Details of IMS are presented in section 2.3.) This is a horizontal network layer providing enablers and support for the service layer; while abstracting away the lower layers for the applications. Enablers are described in section 2.3.8.
4. The service layer consists of applications for users and machine-to-machine (M2M) applications. The idea is that the application developers for the service layer can use IMS to create richer applications: develop these applications faster; deliver, and deploy them at a

lower cost. Essential to the service layer are application servers (AS). These servers host client-server applications, act as back-to-back user agents (B2BUA), or adding enablers for peep-to-peer (P2P) applications.

In terms of the layers presented in the previous chapter, we can see that the above can be mapped in several ways on to the earlier model. The passive network operator will provide the physical transport layer. The communication operator provides the multi-access layer, but may or may not provide the IMS core. In the municipal network case it is beneficial to have the multi-access layer available to all service providers, as opposed to a single service provider providing IMS to other service providers (and to themselves); i.e., this means that the multi-access layer should be open to all service providers – just as the physical transport layer is. Finally, the service providers will provide the services of the service layer.

2.2 Horizontal Services

In an IMS enabled network we can deploy horizontal services. A horizontal service is not the same thing as the horizontal network architecture. Horizontal services allow applications to re-use existing code and enablers (see Figure 3). For instance, sending instant messages is an IMS enabler that can be used by an application in order to easily send an instant message. The application only needs to make a function call to the API specified in JSR 281 (IMS Services) [7]. If the application wants to set up a VoIP call it only needs to make a function call instead of having to build all the headers for the SIP message and open sockets manually.

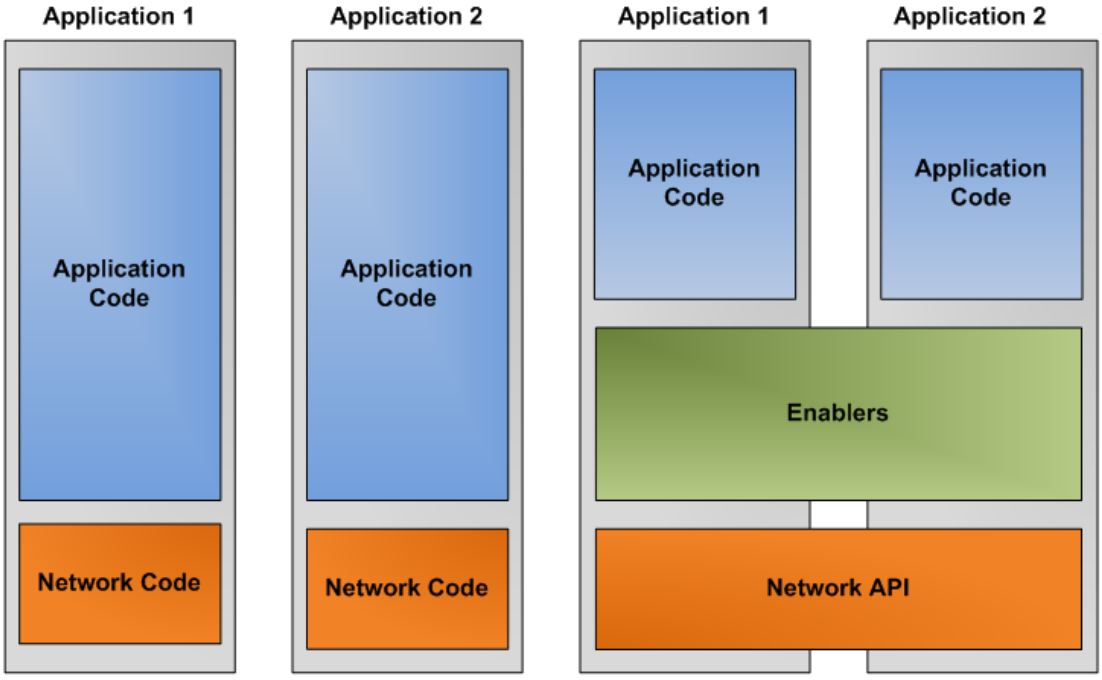

Figure 3: Standard applications **IMS applications**

2.3 IP based Multimedia Subsystem (IMS)

The IP based Multimedia Subsystem (IMS) is an intelligent IP network core designed to deliver multimedia services. It is part of the 3GPP specification for a Next Generation Network (NGN) based upon an all-IP network. IMS was originally specified by 3GPP for 3G networks, but is now

supported by additional actors, including TISPAN [8]. Although IMS was intended for cellular networks, it has evolved into a subsystem for any type of broadband network. To make it easier to integrate IMS into an all-IP network the protocols used by IMS has been adopted from protocols specified by the Internet Engineering Task Force (IETF). IMS consists of a number of network elements and a number of protocols. The basic architecture and major protocols are described in the following sections.

2.3.1 IMS Layers

The IMS network can be divided into several layers (see Figure 4):

Application Plane

The application plane contains the Application Servers (ASs), Home Subscriber Server (HSS), and the Presence and Group Management (PGM) Server with database. (These will be explained in sections 2.4, 2.3.5, and 0 – respectively.)

Control Plane

The control plane contains the I/P/S CSCF and BGCF. (These will be explained in sections 2.3.3 and 2.3.4, respectively.)

Access network

The access network connects the user equipment to the different networks such as PSTN, internet and other IP based networks, and cellular networks.

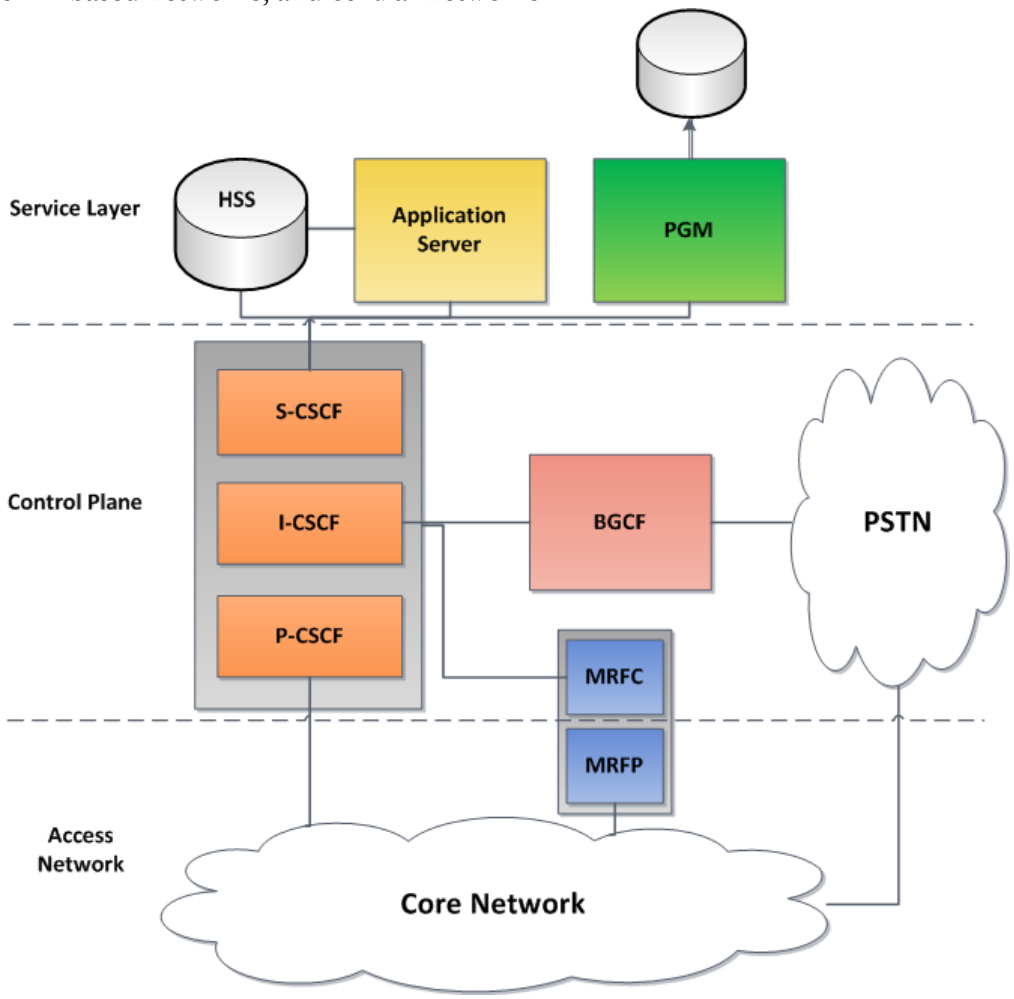

Figure 4: IMS architecture

2.3.2 Session Initiation protocol (SIP)

IETF's Session Initiation protocol (SIP) [9] is used for setting up and tearing down multimedia sessions. SIP builds upon many ideas from HTTP. Similar to HTTP where clients initiate requests and servers can respond with HTML, SIP clients initiate SIP requests. Additionally, SIP user agents can act as clients, servers, or both. As clients they initiate SIP requests and as servers they respond to SIP requests. Because of SIP's adaptability it was chosen as the core protocol for IMS. The most interesting parts of the SIP header are:

Type	A message can be an a SIP request, such as : INVITE, or a response such as: 200 OK
Via	This header specifies the path the message should follow (if there are proxies involved this enables them to specify if they are to be kept in the path for the response).
To	The recipient of the message, often a URI of the form: sip:alice@example.com
From	The sender of the message, also often a URI
Content type	The content type header specifies what type of content is included in the body. The most common value is "application/sdp", indicating that there is a Session Description Protocol (SDP) [10] message in the body.
SDP	Formally SDP is not part of SIP, but is carried inside the SIP message body. SDP describes a multimedia session that the sender and receiver will initiate. This description specifies what ports, protocols, and CODECs the multimedia streams will use during the session.

2.3.3 Call Session Control Function (S-, I- and P-CSCFs)

The Call Session Control Functions (CSCFs) are SIP servers and form the core of IMS. A CSCF handles SIP signaling and initiates/terminates user sessions. The various CSCF's also communicate with the Home Subscriber Server (HSS). There are three types of CSCF:

- **Serving-CSCF** (S-CSCF)
  Each user is assigned an S-CSCF which acts as a stateful SIP server for this user. A S-CSCF uses the Diameter protocol to authenticate the user with the HSS.
- **Proxy-CSCF** (P-CSCF)
  A P-CSCF acts as a SIP proxy server. One P-CSCF is selected as the first point of contact for the subscriber's terminal. Communication between the subscriber's terminal and the P-CSCF is integrity protected using IPsec [11]. The P-CSCF is responsible for checking the correctness of all SIP messages sent by the subscriber's terminal. If the communication between the subscriber's terminal and the P-CSCF is over a limited bandwidth connection, then the P-CSCF can compress the SIP messages to conserve bandwidth. The P-CSCF adds charging information to the SIP header (P - Charging-Vector).
- **Interrogating-CSCF** (I-CSCF)
  The I-CSCF is a home network's point of presence. The I-CSCF identifies which S-CSCF each user is connected to.

2.3.4 Breakout Gateway Control function (BGCF)

The BGCF is used to connect an IMS with a circuit-switched Public Switched Telephony Network (PSTN). A call that is destined for the PSTN is routed through the BGCF to a set of gateways and into the PSTN. The BGCF also handles incoming calls from the PSTN and converts them to IP packets. The S-CSCF routes the incoming call to the IMS subscriber.

2.3.5 Home Subscriber Server (HSS)

The Home Subscriber Server combines the functions of GSM's Home Location Register (HLR) and the Visitor Location Register (VLR). It is a database containing all the subscriber information for this network and information about all visiting subscribers. The HSS communicates with the S-CSCF and I-CSCF using the DIAMETER [12] protocol. DIAMETER is the successor to the earlier RADIUS [16] authentication, authorization, and accounting (AAA) protocol. The HSS utilizes a Uniform Registration Identifier (URI) that is the subscriber's IMS identity. This is similar to a phone number in the PSTN and can be used to reach the user regardless of the service used. If the URI is actually a phone number, then it is called a TEL URI. One important feature of the HSS is to store criteria for triggering of an application server (AS) in the S-CSCF (see section 2.4).

2.3.6 Presence and Group Management (PGM)

The Presence and Group Management (PGM) server is located in the application layer of IMS. The PGM communicates directly with the IMS core and with application servers. The presence enabler in IMS allows applications to learn the subscriber's current status using the SIP SIMPLE presence framework. Having this function available for all applications makes the network more horizontal. Users can create a contact list using the Group Management service and they can store their profile in a PGM server.

2.3.7 Media Resource Function (MRF)

The Media Resource Function (MRF) is a component used when users require media from the network or when several users are in a multimedia conference. The MRF can send the subscriber an announcement, play voicemail, generate a voice prompt, etc. The MRF can also operate as a reflector for a conference call.

There are two components to the MRF: the MRF Controller (MRFC) and the MRF Processor (MRFP). The MRFC acts as a SIP user agent to the S-CSCF and sends commands to the MRFP which handles the actual media streams.

2.3.8 IMS Enablers

Enablers are key features in IMS that are provided to all service layer applications.

- **Messaging**

  In the SIP protocol there is support for sending SIP messages. These messages can be used as a form of instant messaging. This functionality can be used by applications and in combination with presence can easily create services. Note that unlike SMS and MMS messages the body of a SIP message can contain any type of data consistent with its specified "Content-type"; however, the specification does indicate that the size of this message should be relatively small. (There is

an IESG requirement that messages outside of a session have to use a congestion avoidance capable transport protocol.)

- **Presence**

  The presence enabler in IMS allows different applications across the network to know your current status (i.e., presence); this makes the network more horizontal. The presence enabler can also be used to tell others what kind of device you are currently using, so they can initiate the best suited media type for your device and current status. For instance if a user is in a meeting and her cell-phone has registered with IMS, then the most appropriate way of communication would be a IMS message rather than a call. If the user on the other hand is registered on their PC (with high bandwidth) and the subscriber is currently available, then the best session type could be a video call.

- **Push to Talk**

  Push to Talk over cellular (PoC) is similar to using a walkie-talkie, you push a transmit button in order to speak. Push to talk is a half-duplex service, meaning that only one person can talk at a time.

2.3.9 Ericsson IMS in the Box (IITB)

IMS systems are targeted at large deployments, having millions of subscribers. For a Municipality that do not require the capacity to support millions of users — this can become a problem. There is a need for a lower end alternative to the telecommunication grade IMS system as these are typically complex. By using virtualization it is possible to merge several of the servers of the IMS core into a single desktop machine. This will reduce the performance and reliability of the system, but will make it affordable even to smaller customers. There is a balance between reliability and price, and for a small number of users the benefits in price far outweighs the loss of a few nines of reliability [18].

By using virtualization the IMS system is using the exact same code as a large IMS solution. This solution is called IMS in the box (IITB). Using the same code is very important as it enables the IITB system to be used as a testbed for IMS application development and leverages the development costs of the IMS code. IITB offers an advantage over Ericsson's Service Development Studio (SDS) (section 3.2.3) that uses a *simulated* IMS core, rather than the real code. The IITB utilized in this thesis project is Ericsson's real IMS software running in a virtualized environment. The setup consists of a host machine running SLES Linux with VMware [14]. There are two virtual machines (VM) running; VM1 and VM2. A Virtual IP manager (VIP) creates an external IP address for each of the VM's that is different from the host machine's IP address.

VM1 Handles the booting of VM1-2, loading of software on them and the communication between them.

VM2 Runs the S-, P- and I-CSCF and HSS on top of the Telecom server platform (TSP) and uses the Dicos operating system (see section 3.2).

The PGM and XDMS on top of Ericsson Application Server (EAS) run natively in Linux on the host machine.

From the outside the system behaves exactly as a real IMS system, thus users will not notice the difference. There is a virtual IP (VIP) service that from the outside acts as one IP for the virtual machines.

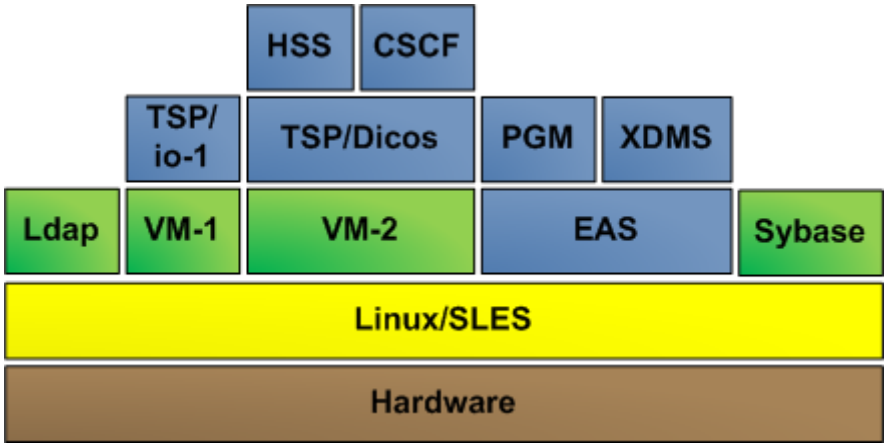

Figure 5: IMS in the box with CoreEAS

2.4 Application Servers

Application server is a broad term. The most basic meaning is simply a dedicated machine running a service for clients. This machine could range from a desktop machine in someone's closet to a cluster server running Google's search engine. Different applications require different kinds of server performance and reliability. Very critical applications require very reliable application servers. Telecommunications operators have a long history of running very reliable severs, as they do not want their telephone network to be unavailable, but this high reliability also comes at a great financial cost. There are several techniques for increasing reliability. These techniques include: failover to backup servers, hard drive redundancy (today usually implemented as some type of RAID), clustering, virtualization, and clouds. In an IMS environment with SIP signaling, application servers also have to understand SIP.

2.4.1 Sailfin Application Server

SUN Microsystems has implemented a Java/EE application server called Glassfish [17] that is widely used on the internet. Glassfish consists of Enterprise JavaBeans [19] and a regular webserver. Ericsson has developed Sailfin [20], an extension of Glassfish with an additional SIP servlet using JSR 289[21]. The commercial Glassfish with Sailfin release is called the Sun Glassfish Communication Server. This server can host dynamic websites with JavaScript, run Java code, and use SIP signaling, a very powerful combination. This server can be used as a back to back user agent (B2BUA) to facilitate communication between two parties.

Application Server Invocation: Stored in the HSS is the user's Service Profile. The service profile is retrieved by the S-CSCF via the DIAMETER protocol. In the Service Profile is an initial filter criteria (iFC) which consist of trigger points and an AS address. When a SIP INVITE reaches the S-CSCF, the S-CSCF examines the user's Service Profile and evaluates the trigger points. If the INVITE matches a trigger point criteria's, then the Invite is forwarded to the assigned AS.

2.5 Service Availability Forum (SAF)

The Service Availability forum [22] is a group of companies working for standardization and adoption of high availability software and management interfaces. SAF works to enable use of commercial off the shelf (COTS) hardware to create (telecommunications) carrier grade systems.

Open SAF [23] is an open source project dedicated to creating middleware based on the SAF specification. Its goal is to increase adaptation of SAF high availability code in commercial products. Many companies are partaking in the Open SAF initiative; both Ericsson and Huawei are representing the telecom sector. Why is the move to a more open service delivery platform so popular? Previously telecom companies developed their own hardware platform, operating systems, programming languages, and software. This becomes very expensive and all components are tied to each other making it hard to change them. Collaborating with many companies from different areas in order to create a better platform is beneficial for all. By using Open SAF companies can:

- Increase openness and avoid vendor lock in,
- Create replaceable components,
- Use open source software,
- Achieve high availability, and
- Achieve scalability.

2.6 Previous Work

In his thesis "Facilitating the adoption and use of the IP Multimedia System" [24] Christos Papazafeiropoulos made several interesting points – based upon his examination of some of the APIs used in Sun Glassfish Communication Server, specifically JSR289 and the IMS service API 281. He created an application and deployed it using Ericsson's Service Development Studio. Based on his measurements of this application, he drew some conclusions about these APIs and the performance of his application in SGCS. One of his conclusions was that the memory consumption in the AS increased by 4 Megabytes for each new user. This large (and linear) increase of memory consumption can be very bad if there is to be a large number of users.

Veronica Kumlin's thesis [25] on open source application servers for IMS examined different application servers. Her conclusion was that there are several open-source application servers on the market, but they are generally not well documented and supported. The AS's investigated all have SIP functionality, but this does not say much about how they run other code, while SGCS runs EJB and servlets. It could be interesting to look at what other options are available in the market and how other organizations implement their IMS application servers.

In their thesis "Experiences from Simulating TSP Clusters in the Simics Full System Simulator" [27] Emil Erlandsson and Olle Eriksson try to develop a simulator for the TSP platform. This is not entirely successful, but gives a lot of insight into the TSP platform and how complex service platforms can be. The problems they had also show how successful the IITB virtualization is.

3 Service Platforms

A service platform is a complete solution for deploying applications, from the hardware all the way to the management of the software. It functions as an applications server, but has much more functionality. This service platform makes sure that the applications are running reliably and are always accessible; while at the same time the platform should be easy to manage and configure.

3.1 Telecom Service Platform (TSP)

The telecom Service platform [28] is one of Ericsson's standard platforms. This architecture is shown in Figure 7. Both the CSCF and the HSS runs on this platform. It is a proprietary platform developed solely by Ericsson from the hardware all the way to the TelORB middleware (TSP clusterware), the only exception is some nodes running Linux as their operating system. TSP employs the same hardware setup as the Open Multimedia Platform (see section 3.2) with applications distributed over several traffic processors (TPs) and controlled by redundant input/output nodes. The similarities between the systems are striking; the differences lie in the way they are implemented. TSP uses Ericsson's own operating system Dicos (TSP kernel). Dicos is specially developed for real-time telecom applications, but the drawback is that it locks the application to the platform. The TSP hardware was developed by Ericsson and is less replaceable than its OMP counterpart (as OMP uses more easily updateable blade servers). Figure 6 shows how a TSP cabinet could look like, large and takes a lot of space.

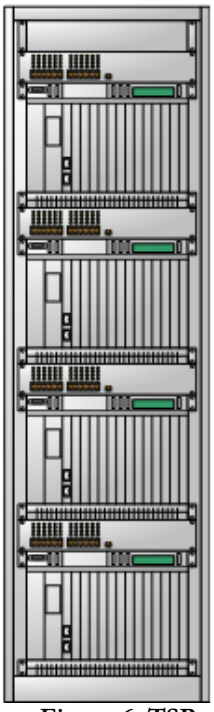

Figure 6: TSP Cabinet

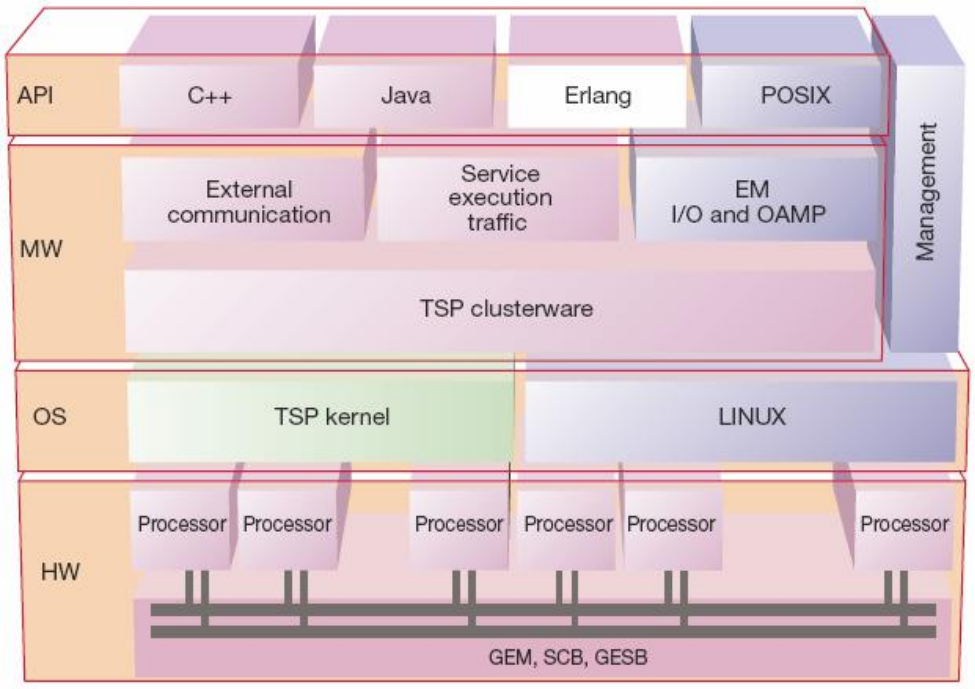

Figure 7: TSP Architecture, Copyright Ericsson AB (used with permission)

3.2 Open Multimedia Platform (OMP)

The Open Multimedia Platform [26] is an implementation of the Open SAF standards (section 2.5). It is a horizontally layered architecture with several components, similar to the SAF specification. It consist not only of Ericsson developed components, but is a mix of Ericsson, Open SAF, and open source software running on COTS hardware (see **Error! Reference source not found.**). By using OMP instead of the older Telecom Service Platform Ericsson does not need to develop all components themselves thereby reducing cost & time to market, and making it easier to find the right competence. The hardware will be a blade server with up to twenty blades, each running an AS. TSP SAF provides load distribution, failover, and management functions for the cluster. An overview of this architecture is shown in Figure 8.

OMP was originally though of as a platform for Ericsson's Multimedia department internal products such as IPTV and Business communication suite. There are roughly 10 applications that are developed on the Java platform and have high availability requirements. Due to these applications there is a possibility for economy of scale; which is of key importance for platforms. In comparison, on the internet there is an abundance of applications and solutions. When thinking about municipalities and their services such as e-health and e-government many of these applications will not be developed by Ericsson. If OMP really is to achieve economy of scale and increase margins it needs to support applications in addition to Ericsson's and be a favorite service delivery platform for third party products.

In a municipal network setting there is a need for low cost **but reliable** application servers. By using OMP with an AS based on Glassfish it will be easier for the municipality to find competent personnel to develop and deploy new applications for e-government and e-health. If all inhabitants of a community where to vote on-line in an election, the service would have to be very robust and very secure.

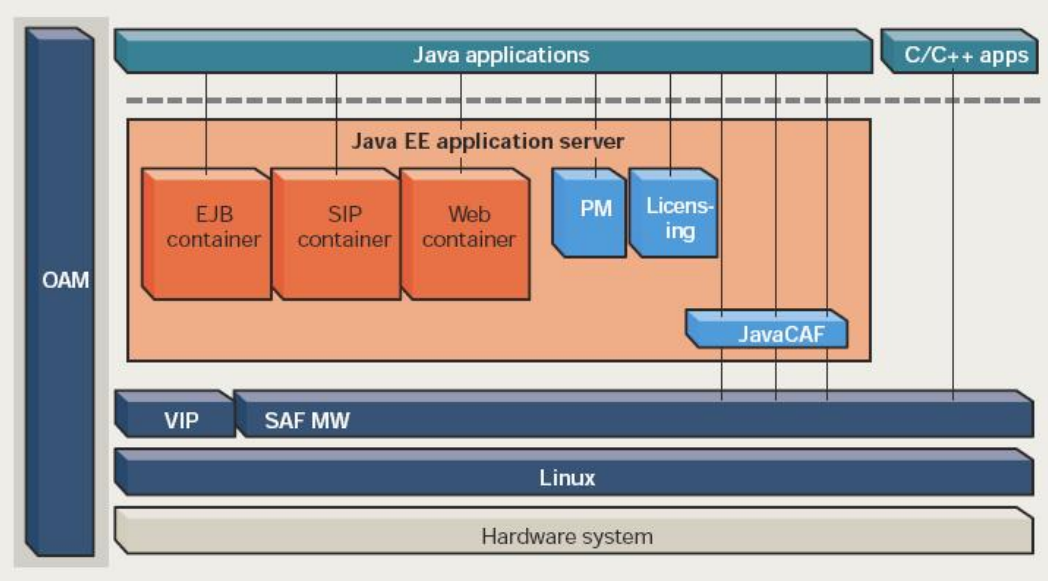

Figure 8: OMP Overview, Copyright Ericsson AB (used with permission)

3.2.1 OMP Components

OMP is not a single product, rather it consists of many different software and hardware components (see Figure 9). It has a horizontal layered structure with well defined interfaces in order to be able to replace components with newer ones and to avoid vendor lock-in.

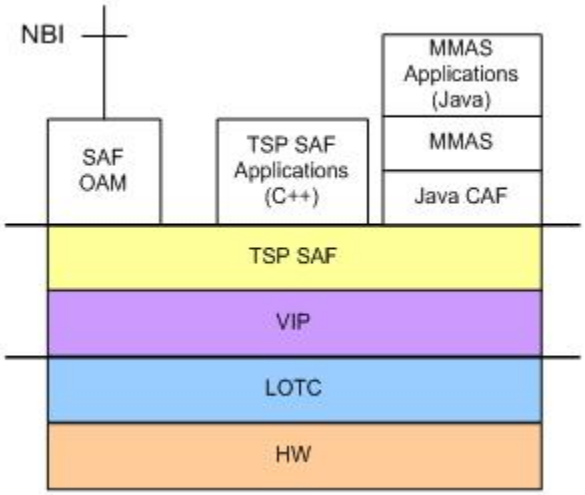

Figure 9: OMP Components, Copyright Ericsson AB (used with permission)

3.2.1.1 Multimedia Application Server (MMAS)

The main component of the Multimedia Application Server (MMAS) is the Java/EE application server SGCS. SGCS uses the facilities defined in JSR 289 for SIP communication with the IMS. In order to interact with the TSP SAF layer there are several API's that are used in the application code in order to exchange information. This could for instance be alarm handlers or configuration management information. In a cluster configuration the domain admin server (DAS) will run on the system controller nodes (SC) and instances of the application will run on the payload nodes (PL). The architecture of the MMAS in a clustered system is shown in Figure 10.

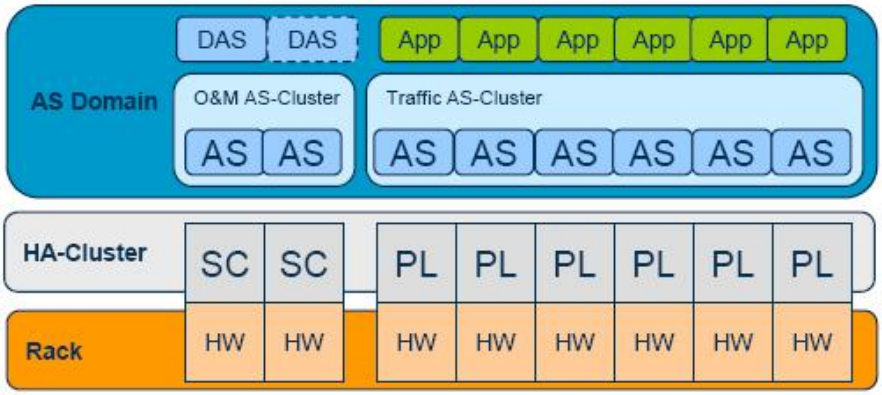

Figure 10: MMAS in a clustered system, Copyright Ericsson AB (used with permission)

3.2.1.2 Java Common Application Features (Java CAF)

Java CAF takes the TSP SAF interfaces and turns them into Java-useable interfaces for the MMAS. Java CAF on a single JVM is shown in Figure 11. Java CAF:

- Provides High-Availability for java applications through AMF. This can be achieved by using JSR 319: Availability Management for Java [13].
- IMM and NTF functionalities through Java Management Extensions (JMX).
- Connects native Java logging function with the TSP SAF logging function

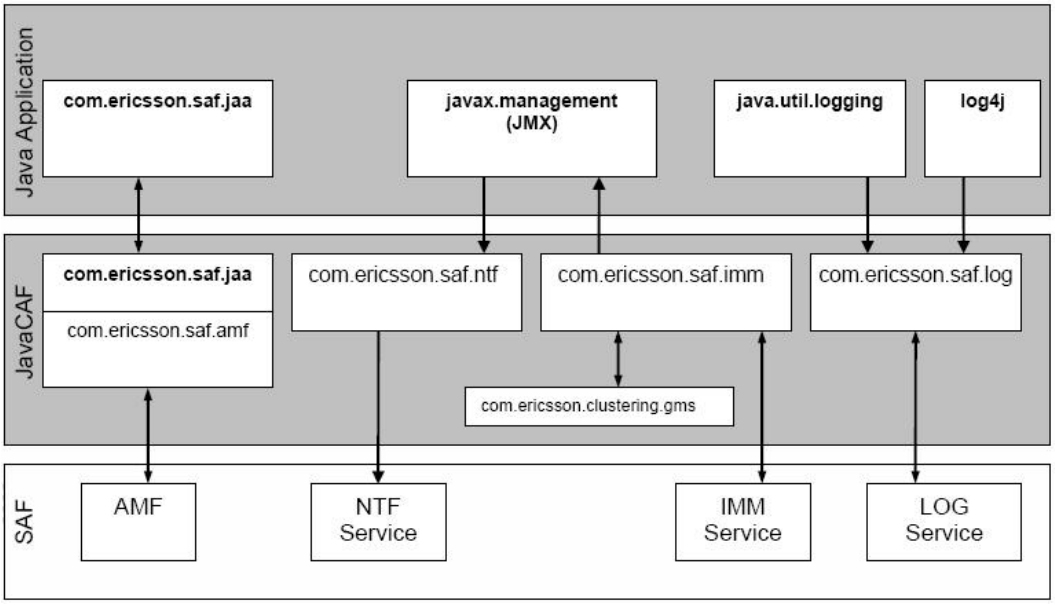

Figure 11: JavaCAF on single JVM, Copyright Ericsson AB (used with permission)

3.2.1.3 TSP SAF

The TSP SAF middleware was designed to provide high availability, scalability, openness and facilitate replaceable components. TSP SAF provides interfaces standardized by the Open SAF consortium. The standardized interfaces are:

Availability Management Framework (AMF)	This service coordinates resources within a cluster to ensure no single point of failure.
Cluster Membership Service (CLM)	Provides applications with information about the availability of different nodes.
Information Model Management Service (IMM)	Via the IMM service managed objects can be created, deleted, read, and modified.
Log Service (LOG)	The LOG service forwards and stores log entries for applications.
Notification Service (NTF)	NTF provides APIs for reading and subscribing to alarms, state change, object events, attribute changes, and security

alarms.

Check Point Service (CKPT)	CKPT monitors and propagates the state of the cluster nodes so the state can be replicated in the event of a failure.
Event Service (EVT)	A communication mechanism based on the concept of event channels.

3.2.1.4 Virtual IP (VIP)

VIP is a single IP address for the entire cluster. External clients do not need to know the actual IP addresses of nodes in the cluster, these clients are transparently redirected to a node inside the cluster. The VIP software attempts to distribute requests evenly across the cluster. By default the load sharing algorithm used is round robin.

3.2.1.5 Linux Open Telecom Cluster (LOTC)

Linux Open Telecom Cluster (LOTC) is a hardened SLES Linux distribution with cluster support and a RAM based root file system to support diskless nodes.

3.2.1.6 Reference Deployment Architecture (RDA)

The Reference Deployment Architecture (RDA) to support an OMP deployment is COTS hardware and network components. The use of COTS components leads to lower prices and shorter time to market (as there is less need for in house development).

3.2.2 High Availability (HA)

The availability management framework (AMF) provides high reliability for the applications. In order to do that, the applications need to be structured into:

3.2.2.1 Components

Components are the smallest parts of the application that the AMF can perform error detection, recovery and repair on. There are three ways to monitor components:

1. Internal active monitoring: Components supporting this are called SA-Aware components. The components include code using the AMF API to monitor its own health and respond to heart beats. To support this, an application has to be developed with the AMF API in mind.
2. External active monitoring: This is used when the components don't have any HA characteristics. A SA-Aware monitor has to be developed that will asses the component by sending service requests. This way non-HA applications can be deployed on a TSP SAF cluster.
3. Passive monitoring: This consists of operating system features to detect the health of the application for example the death of a process.

3.2.2.2 Service Units (SU)

Components are grouped into service units. Together components form a SU that provide a higher level service. The SU can have different states:

HA State	Active, Standby, Quiesced [15]
Readiness states	In Service, Out of Service, Stopping

3.2.2.3 Service Groups (SG)

Several Units together form Service Groups. The SG defines with what redundancy the SU's will run. Some examples of redundancy models are:

2N	Each active component has its own standby
N-Way	One component can take several active or standby assignments at the same time.
N-Way Active	All components are active.
No Redundancy	All components are active with no standby.

3.2.3 Developing on OMP

OMP [26] is designed to be a reliable platform for deploying services, but experience from development on early versions has shown that making applications run on the OMP framework is not easy. To make it easier to develop, package, and deploy application Ericsson is working on an OMP test environment. There are three steps to application development:

1. Development in the Ericsson Service Development Studio (SDS) [30] environment where developers can program in Java and use high level IMS APIs– see Figure 12. SDS includes a simulated IMS environment with SCSF, HSS, DNS, and BGCF functions that can be used to test IMS applications. SDS also includes the SGCS application server. Since SGCS is based on Glassfish, which is a widely used Java/EE AS there is an abundance of documentation and tutorials for developers (see for example [31], [33]).

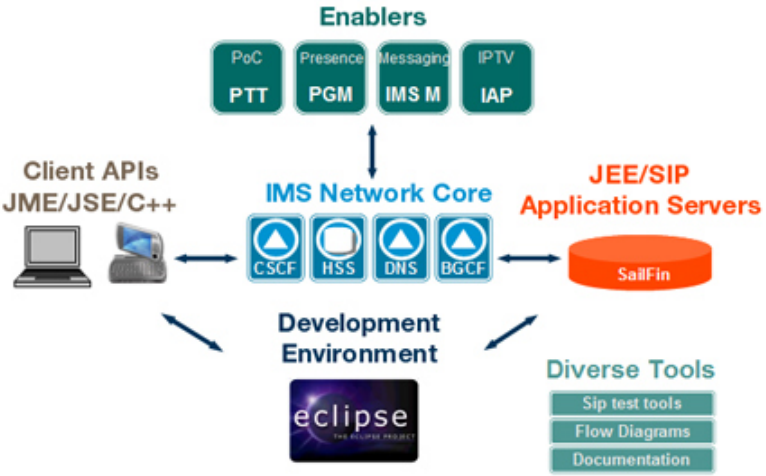

Figure 12: SDS4.1 Overview, Copyright Ericsson AB (used with permission)

2. Develop/Test/Deploy applications on DIXI 2.0 (DX2.0), Ericsson's new virtualized OMP environment. DX2.0 comes with an integrated plug-in for Eclipse to help in development and deployment onto the virtualized OMP system. (See Figure 14) This target environment is typically run on a PC. The virtualized environment is based on Ericsson's TSP SAF Simulator (ESSIM), but is a single node configuration instead of a cluster to limit the performance requirements of the underlying hardware and software. (See Figure 13)

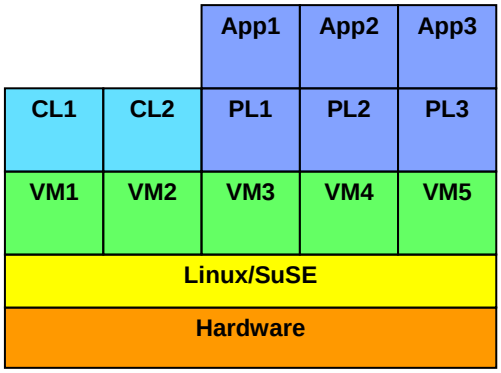

Figure 13: Overview of ESSIM

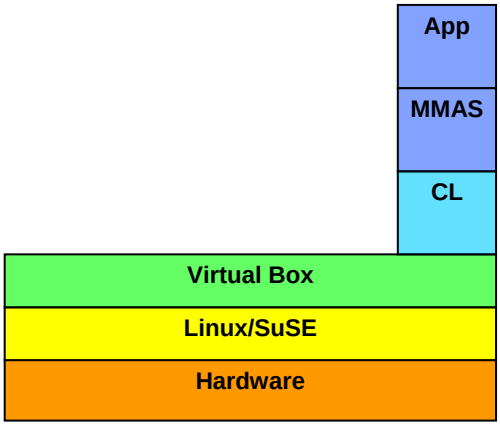

Figure 14: Overview of DX 2.0

3. Finally, the goal is to deploy the application on a real OM Platform. For internal Ericsson developers this is not a problem as there are several deployments of OMP internally. For external developers this can pose a problem and alternatives have to be investigated.

Developing applications for OMP is still pretty new, thus it is uncertain if all necessary documentation exists or that all the specified functionality actually exists. This is especially true of the OMP Development Platform and virtualized environment; as this software is still under active development and has not yet been used by developers. As part of this thesis project we want to see if there is any way to integrate the OMP development environment into SDS, as this is the standard Ericsson service development platform.

3.2.4 Alternatives to OMP

There are several alternatives to OMP and there exist a number of different techniques for achieving high availability; for example, Ericsson already has several other platforms (such as TSP). There are generally two ways of achieving high availability: fault tolerant servers and software clustering. There is a move in the industry towards software clustering as this is seen as a more open, flexible, and cost effective alternative. Open SAF (section 2.5) is one such initiative.

VMware vSphere [35] uses virtualization and software clustering to achieve HA. VMware Fault Tolerance (FT) runs just like an ordinary VM cluster, allowing several VM's on different hardware platforms controlled from a single point. You can indicate that you want FT on a VM by simply clicking on it, an exact copy of that VM will be started on another physical machine. By using a technology called vLockstep, all commands issued and network traffic sent will be mirrored on the secondary VM. The secondary VM will use a heartbeat signal to try to determine if the primary VM is operational, if not, then the secondary VM will instantly take over as the primary VM. Because it has the exact same state as the primary VM, both with regard to disk, memory, and network states there is no disruption of any kind and the switch is completely transparent for the users. This way of implementing HA does not make any requirements on the software being HA aware or requiring it to be modified for running on a VM; this is a great advantage. The application can also run on any operating system that can run in the VM. The major drawback compared to OMP is that the heartbeat can only detect hardware malfunctions, the switch will only be performed if the primary VM's host hardware malfunctions. Whereas, OMP can detect problems on the application layer, such as when an application has crashed, frozen, or failed in any other way -- even when the hardware is still working.

4 Teleconsulta: e-health application

4.1 Introduction

Teleconsulta is a medical teleconsulting application to enable doctors to have videoconferences with their patients, share images, and manage patients in queues. The system is developed for desktop PCs. Teleconsulta is a proof of concept application created to facilitate growth of the IMS business. The application is being developed in a collaboration between Ericsson, Telefonica, and the Polytechnic University of Madrid (UPM). By collaborating with partners that have real world requirements the application should demonstrate valuable service enablers.

In a municipal network setting we assume that there is a FTTH connection to every household, meaning high bandwidth for all users. Communication operators in the network will provide an IMS core with all necessary IMS features. In this setting the Teleconsulta application can be used to lower the load on doctors and to make it easier for patients to get in contact with their doctor(s) without needing to travel to the hospital and wait there. If the issue is not an emergency, then patients can consult their doctors via videoconference. The doctor can decide if the patient needs to go to the hospital or the doctor might prescribe a prescription for the patient on-line, so that the patient can simply pick up the prescribed medication at their local pharmacy. After being at the hospital, for example to get an x-ray, the patient can share their images online with their doctor and discuss the results without the need to travel to the hospital again. If there is a need for the opinion of a second doctor or if the patients want a relative to participate in the conference, the conference can be expanded to include multiple users.

4.2 Architecture and Components

The project consists of two parts; (1) the Teleconsulta proof of concept application and (2) the Service enablers.

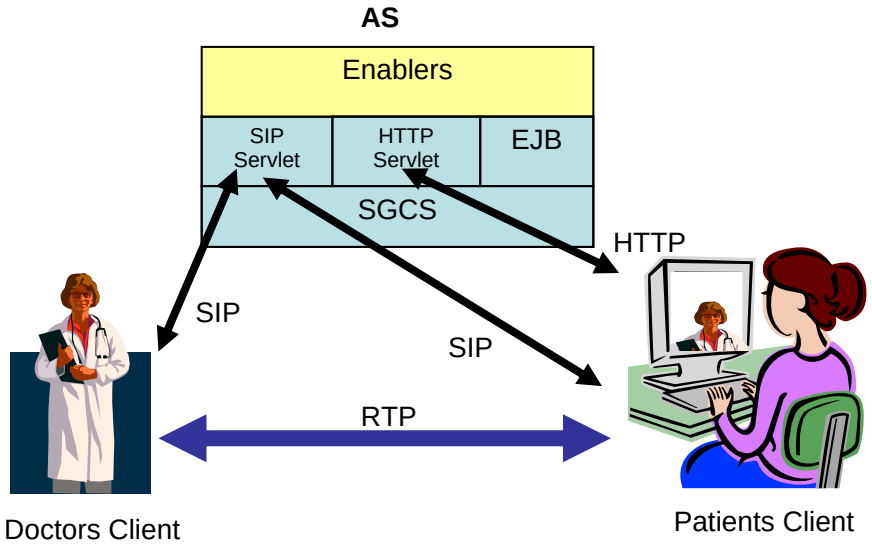

Figure 15: Overview of Teleconsulta, excluding IMS

4.2.1 Teleconsulta Client Application

The Teleconsulta client application is based on SIP-Communicator [29] which is an open source project. There is a simple GUI to use all the functionality developed in Java SE. Using the GUI the patient can manage a contact list with presence, make and receive video calls, and send messages. The doctor can make video calls, manage a contact list with presence, manage the patient queue, share images, and send messages. The focus of the project has **not** been on the clients, but more on the service enablers.

4.2.2 Service Enablers

The enablers are generic and could be used for other applications, such as a call center or help desk. They are designed to be reused for other applications, thus creating a richer service layer (figure 14). The enablers are Java servlets that run on the Sailfin applications server. All the enablers run on one application server which could be called the "IMS enabler AS". The entire project has been developed and tested in the SDS 4.1 environment. The enablers that have been developed are:

- **Group management**
  The group manager exposes the PGM's XCAP interface as a WebService to third party applications. The group manager can create/delete groups, add/delete members, edit group capacity, and retrieve group member lists. This enabler is the base for the other enablers which will make use of the group manager.
- **Virtual Queue**
  The virtual queue gives applications a mechanism to queue users that request a limited resource. The users in the queues are notified about their position in the queue and their currently estimated waiting time. The doctor can request the queue, then choose to talk to the person at the front of the queue or could pick another patient from the queue (for example, if this patient has a scheduled appointment). An alternative solution would be to automatically insert patients with an appointment appropriately in the queue.
- **Virtual waiting Room**
  Applications can provide users with virtual waiting rooms where users can use multimedia resources while they are waiting in queue. The media could be related to the specific service, maybe the latest medical journal. The content could also be adapted to the specific user, the system remembering the user's profile. If the user wants to look at other content outside of the waiting room he is of course free to do so.
- **Synchronized Image Sharing**
  During a videoconference between parties someone might want to show images to the other user(s). This could be your vacation pictures, PowerPoint slides, documents, or as in the Teleconsulta case medical images. All participants can see the image and discuss them. A possible extension could be enabling users to draw on the pictures and make notes.
- **Video Conferencing**
  The video conferencing enabler gives users the ability to participate in audio and video conference. The group manager is used here. All users in a group can participate in a multi-conference call. The multi-conference makes use of the MRFC/MRFP in the IMS network; however, due to lack of these nodes in the thesis lab setup - this enabler will not be used. However, a point to point video call is possible.

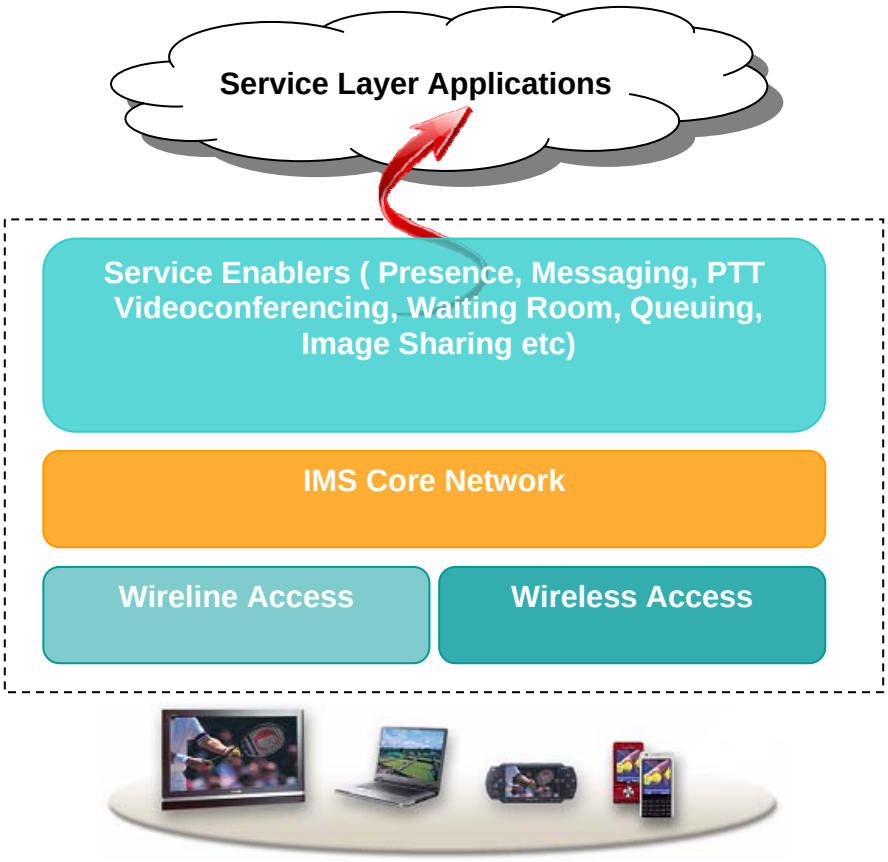

Figure 16: Enablers Overview

4.3 Use case

We assume that a patient wants to consult their doctor about a medical condition; however, this doctor is busy when the patient places their call. The signaling is shown in Figure 17.

- The AS with the enablers subscribes to the doctor's presence.
- When the doctor comes online, he or she registers with the CSCF.
- Doctor creates a VirtualQueue and the VirtualWaitingroom in the AS using the enablers.
- The doctor initiates a call with another patient and as a result is currently busy. (not shown)
- The patient calls the doctor using his or her SIP client.
- A trigger in the CSCF sends the SIP INVITE to the AS instead of the doctor, since the doctor's presence indicates that he or she is currently busy.
- The AS creates a HTML page and sends a 480 Temporary Unavailable back to the patients SIP client including an HTTP link which the user can choose to open, putting them in the virtual waiting room.
- In the virtual waiting room which is a website the user is placed in a queue and learns that the doctor is busy. While waiting he or she can see the remaining estimated waiting time and browse multimedia content.

- When the doctor changes his or her presence from busy to available, his or her SIP client checks with the AS to learn who is at the front of the queue (using a SIP MESSAGE). The doctor can then invite the patient to a videoconference.

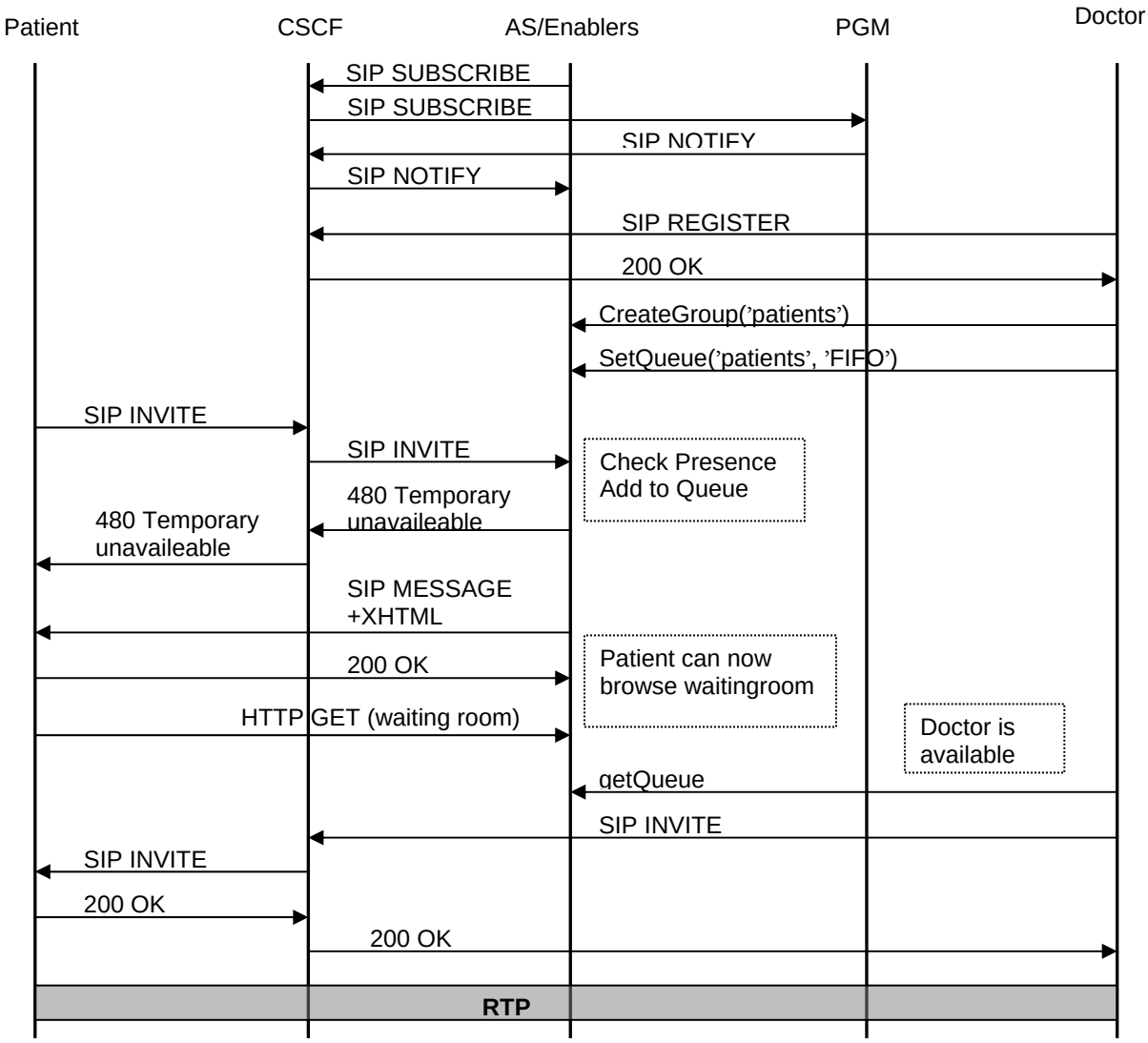

Figure 17: Patient call when doctor is busy, simplified

5 Evaluation procedure

The purpose of the evaluation is to gain knowledge of how the systems and applications mentioned function and their ability to work together. For testing the Teleconsulta application with enablers will be used. This application is very interesting from a municipal network point of view. This application will be evaluated from a functional and conceptual perspective, as opposed to reviewing the actual coding itself. As this application relies on an IMS system, we will use Ericsson's IITB system (see section 2.3.9). Documenting that the virtualized solution works well with all IMS applications and that new functionality can be included will also be useful for Ericsson. Showing that IITB could be a suitable IMS system for municipalities and that it works well with videoconferencing applications would be an extra benefit. Note that the focus is **not** testing OMP's performance in a full scale deployment on target hardware. Thus we focus on the following questions (note: the full list of evaluation criteria are listed in section 5.2):

- How well does the software match its description?
- How well does OMP work as a service platform for third party municipality applications?
- How easy is the development environment to use? Specifically, how well documented is the development environment and all the APIs necessary to develop the Teleconsulta application?

5.1 Evaluation Environment

There are several stages of the evaluation. Each of the stages focuses on different components. These stages and their focus are:

1. Enablers + IMS	The enablers will be deployed on Sailfin using SDS 4.1 running on a desktop machine. A virtualized IMS system will be set up to support IMS features. The AS containing the enablers will be provisioned and authenticated in the IMS system
2. Teleconsulta Clients + Enablers + IMS	This stage starts with the configuration, the provisions users. This will require installing the patient/ doctor applications and making calls, and using the queue and waiting room.
3. ESSIM	Install an ESSIM virtualized cluster environment on a lab PC running SLES Linux as the OS. This is the early version of the OMP TE and has been reported to not be user friendly. Use ESSIM to package, deploy and run a sample* application. This will be used as a reference point later.
4. OMP DX 2.0	Get and install the new OMP 2.0 test environment with Eclipse plugin. Run sample* applications and compare to ESSIM.
5. OMP DX 2.0 + Enablers + Clients + IMS	Make the application HA-aware either by changing the code or encapsulating it. Package and deploy it using OMP TE. Run the client application through the AS with IITB.

5.2 Evaluation Criteria

The evaluation criteria can be split into several clusters corresponding to the stages of the evaluation described in the previous section, thus the evaluation criteria will be:

- Enablers + IMS
  - Is this procedure well documented?
  - How difficult is IITB to set up and use?
  - Are IITB and the enablers compatible?
  - How much configuration has to be done?
  - Is there a limitation on what applications can be run on IITB?
- Teleconsulta Clients + Enablers + IMS
  - How easy is the installation and setup of the solution?
  - Do the clients use correct SIP signaling and are these clients compatible with IMS?
  - Is there a performance limitation of the AS running the enablers (i.e., how does the performance scale)?
- ESSIM
  - How easy and clear is the installation procedure?
  - How useful is ESSIM for developers?
- OMP DX 2.0
  - Has the installation process improved over ESSIM?
  - Is it easy to package and deploy projects?
  - Does OMP DX fulfill the role it is supposed to?
  - What features of OMP can be used in DX2?
- OMP DX 2.0 + Enablers + Clients + IMS
  - Is it possible to port an already developed application to OMP?
  - Is it possible to use OMP DX in this setup?
  - What is the performance in this virtualized environment?
  - Does the application benefit from using OMP?

6 Evaluation and Analysis

The analysis and evaluation will be structure into the same stages as used in the previous chapter.

6.1 Enablers + IMS

Initially the enablers where installed on a regular Windows PC†. It was very easy to deploy the enablers in Sailfin through SDS.

IMS in the Box (IITB) was used as the IMS system. Several commands needed to be entered to install and setup an IITB server. To facilitate this process and make it easier and faster a Start/Stop/Restart bash script was made. By running the commands as scripts the process is much faster, there is less chance of doing something wrong, and the user does not need to supervise the whole process (this is advantageous as the complete process takes quite some time - ~20 minutes). The script can be seen in appendix A.5.

The group management enabler creates and modifies groups. The other enablers utilize the group manager to update the PGM. The interface to change the groups is not SIP, rather it is the XML Configuration Access Protocol (XCAP) built on top of HTTP. Initially, when the enabler tried to access the PGM through the XCAP interface there were authentication problems. Because not everyone should be able to access the PGM through this interface there is a set of authentication procedures. Regular users are registered with the CSCF and have an authenticated user identity which can be used to authenticate them to the PGM, but an AS does not have such an identity. As the focus of the application was not authentication, there was a temptation to disable authentication altogether. Although the developers in Spain had their own "real" IMS system with some pre-configured settings for the AS, the setting to use in our lab where hard to come by.

In order to learn the correct settings for configuration of the authentication, contact was initiated with the developers of PGM. They said that certain settings needed to be set in the SPA console of the IMS system, a web-interface where PGM settings can be configured. However, upon inspection these settings where not available on the system. We assumed that this could be due to using the IITB solution. However, the IITB developers claimed that they ran a regular PGM, exactly as the official release and they indicated that we should talk to PGM personnel. After some discussions back and forth it was revealed that there was another way to edit the settings that revealed settings hidden by the SPA console. Most settings in the SPA console are marketed as runtime-editable, but changing these settings requires a restart of the PGM.

Due to difficulties in the Teleconsulta-IMS in the box integration it was decided that we would spend one week at Ericsson's office in Madrid collaborating with them to solve these issues. They ran their application in a configuration with a simulated CSCF and HSS from SDS, the PGM and MRFC were running in TSP cabinets, and the enablers were running on Linux servers. In total, this configuration had 2 TSP cabinets, 2 Linux servers, and one desktop PC for SDS. Because the core was only a simulation of the real CSCF, the system did not behave as it would if this would have been the real CSCF. This difference occurs because SDS has simplified communication and trigger rules that in some cases differ from the real CSCF. In order to be able to demonstrate the application

† In the tests conducted in this project this PC was a HP laptop model 2510p. The processor was a Intel Core2Duo with a maximum clock rate of 1.33 GHz. The computer was equipped with 2 GB of memory and 200 GB of SATA 2 disk space.

to external people, a decision was made to run the real CSCF code and to configure the system such that the entire system fit into IMS in the Box. The goals of the week were to:

1. Install IITB on their target server.
2. Successfully solve all IITB/Enabler integration issues.
3. Configure Trigger data in the HSS.
4. Evaluate the possibility of integrating the MRFC component in IITB.

The desired configuration is shown in Figure 18 and Figure 19. Details of each of these goals and how they were achieved are summarized in the following subsections.

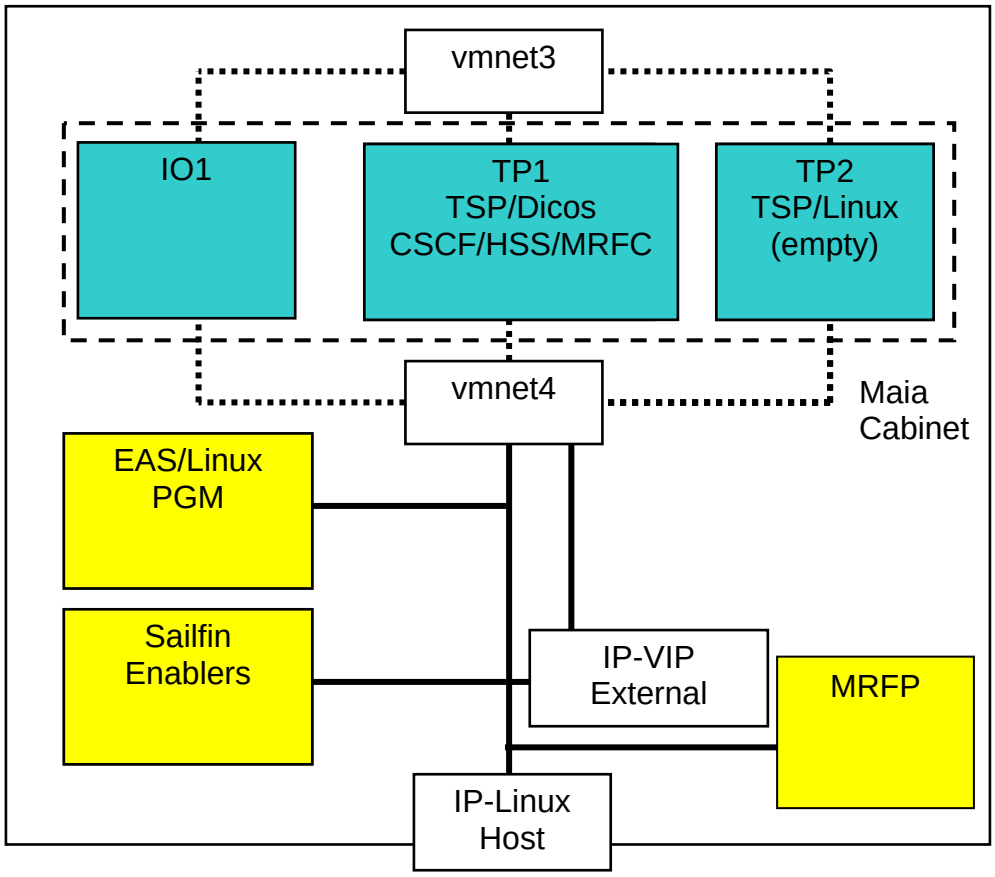

Figure 18: Target IITB + Enablers Architecture

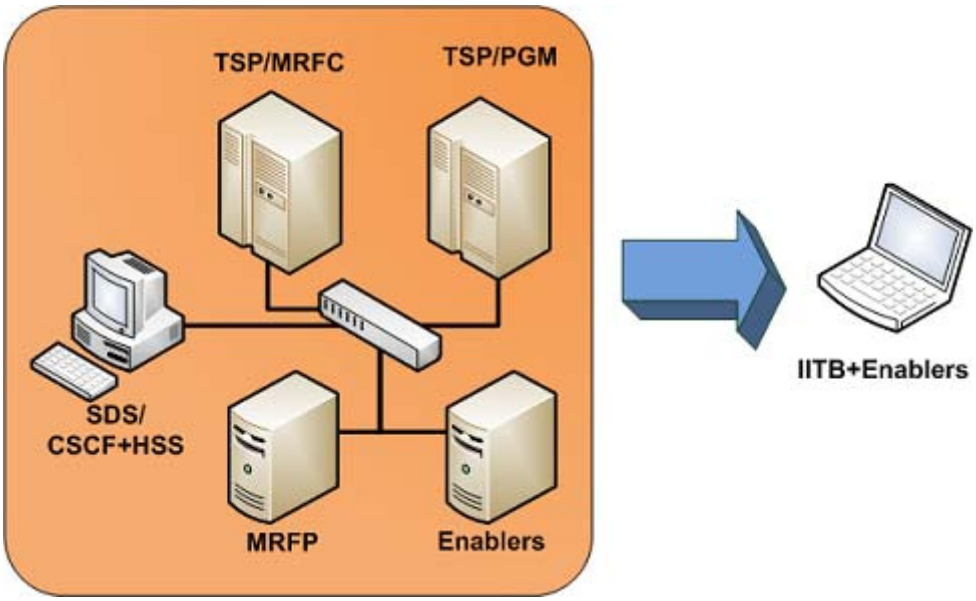

Figure 19: Physical view of Components virtualized in IITB

6.1.1 Install IITB on target server

The installation of the new version 2.1 release of IITB went very smoothly. The total installation took roughly two hours. Using CounterPath's X-Lite SIP client [36] it was confirmed that registration, messaging, calling, and presence worked well. This installation time is a clear improvement from earlier version where it could take up to a week for an inexperienced user to perform a comparable installation.

6.1.2 Successfully solve all IITB/Enabler integration

By having the PGM running on the Linux host, instead of inside the virtual machine TP2, the VIP no longer forwards requests to it in the exact manner as done in the real TSP cabinet, thus traffic had to be directly addressed to the PGM. The PGM also needed to have extensive configuration of the Aggregation Proxy in order to work. The first problem that was encountered was that the XCAP AP ran in Front End (FE) mode, hiding a lot of settings, including how to perform authentication (as described earlier). We had to use a preference editor tool called ped.sh in order to see the current configuration and change the running mode from FE to AP. Following these changes we needed to reboot the EAS/PGM. In our configuration the AS is in a trusted zone (in fact it is running in the same PC), therefore we turned off global authentication and added a superuserid that all requests from the AS would be assigned as their identity. The XCAP Proxy talks in turn to the XML Document Management Server (XDMS) where the groups and presence information are stored. The default settings needed to be modified through the SPA console. Some of the most important settings that needed to be changed were the service mappings and XcapPSITemplate. We installed all the enablers in two different Sailfin domains on the Linux host. A third domain was used as a repository for the images used in the ImageSharing enabler. Running all these webservers is performance demanding for the computer, but as the traffic on the test system will be limited this was not expected to be a problem. The logical configuration of these servers is shown in Figure 20.

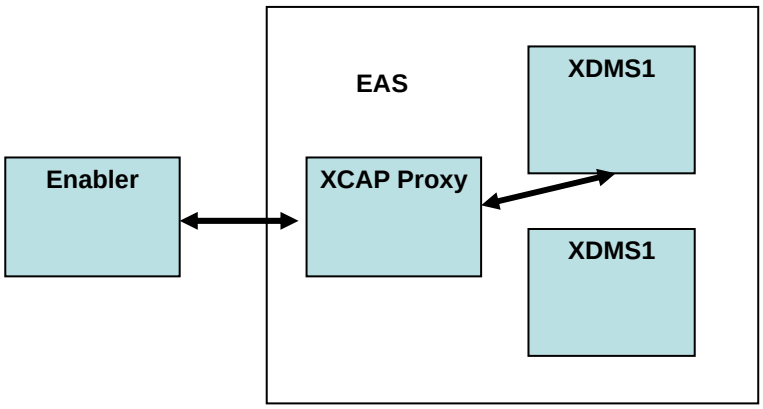

Figure 20: XCAP and XDMS

6.1.3 Configure Trigger data in the HSS

In order for the enablers to work, a user who is a doctor needs to have a trigger set in the CSCF to send all SIP INVITEs for this user to the enabler AS. Previously the IMS core had been simulated in SDS with some very easy and simplified tools for handling triggers. However, the IITB features a real HSS, thus there where no similar shortcuts. Upon registration of the user the CSCF requests the user's service profile from the HSS: This service profile must contain the relevant triggers with the appropriate priorities. Each trigger has several conditions that need to be met in order for this trigger to be true -- as a result invoking the action associated with this trigger. The trigger also contains information that the CSCF needs in order to forward the request to the correct AS. To provision a user with the correct service profile, we first had to create triggers. A trigger was created named `HSS-TriggerPriority=201` that would trigger on a user terminated INVITE and would forward this INVITE to our AS. Next we associated this trigger with a `HSS-ServiceProfileId` called `VirtualQueue`. There can be several ServiceProfeId's. These IDs are not connected associated with specific users. The ServiceProfileId will subsequently be saved in a subtree: `Hss-ServiceTypeId=CallQueueService`. The final step in this process is to associate each specific user's ServiceProfile with the correct ServiceProfileId by editing `HSS-ConfiguredServiceProfiles=VirtualQueue`. The resulting user data in the HSS is shown in Figure 21.

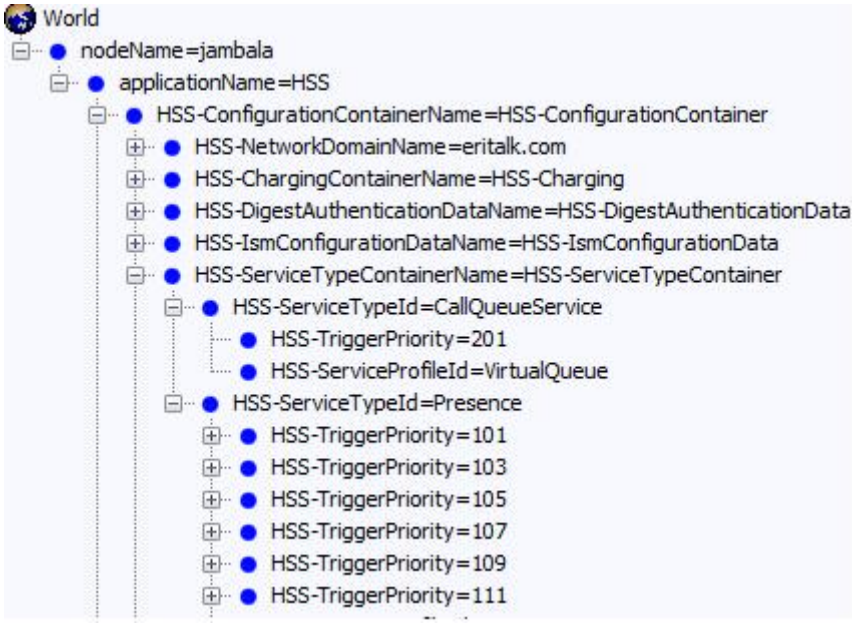

Figure 21: View of the ldap tree in HSS

Creating users with the creating ServiceProfieIds was simplified by using scripts in IITB. When entering the command `createUserHss <name> <service1>…<service n>` a `ldif` file is created that can be added to the LDAP server in IITB as shown in Figure 22 by using the command `addUserHss <name>.ldif`. When we tested this by calling the doctor as another user, the invite was forwarded by the CSCF to the AS. However, because the AS is collocated on the same Linux host and because of the configuration of the routing tables the AS replied to the CSCF on a virtual network interface (specifically: vmnet4) instead of on the public interface as one might expect. Due to the SIP via headers not matching and the sending interface's IP address not being a trusted AS the message was dropped! In order to make a quick fix for this we added the Linux host's vmnet4 IP address as a trusted AS and changed the trigger to send the INVITE to the virtual interface instead of the public network interface, this way the via headers in the SIP packet would be correct. In order for the videoconference to work, the videoconference enabler also needs a trigger. This trigger has the virtual user ORIGINATING as its trigger. When the doctor initiates a call with a patient the invite is routed through the videoconference enabler, thus making this enabler aware of the two conference participants.

Figure 22: View of the user data in HSS

6.1.4 Evaluate the possibility of integrating the MRFC component in IITB.

The MRCP is a IMS node that runs natively on the TSP/Dicos operating system. This is the same OS as used for the TP1 virtual machine in IITB. Therefore it should not be impossible to change the configuration of the TSP cabinet to include the MRFC component. Because IITB automatically does all the installation it was initially expected that it would be hard to add the MRFC component. But after investigating the scripts it was revealed that the install scripts look in the /etc/jimldap/schema/common.schema.template file and we appended the path to our schema files here. All installable components have a schema file that determines how they behave and how they are to be installed. All components in TSP are stored in the IO1 virtual machine in the `/usr/lib` directory. By adding the MRFC schema to this directory the install scripts recognized it and automatically configured it during the installation. The `epct` command reads the install configuration and creates a loading group for each application. The Traffic processor(s) will subsequently boot and load their configuration from the relevant loading group into memory. By default the MRFC listens to SIP port 5060, but this port was already used by the I-CSCF so we modified the schema file to use port 5070. During the install this port was configured in the VIP. To configure the system in this manner the following files had to be added to the IO1 directory:

```
/etc/cabinet/preconf_mrfc.dicos
/etc/cabinet/prov_mrfc.dicos
/user/lib/mrfc/epct/e200mrfc.dicos
/user/lib/mrfc/schema/mrfc.schema
/user/lib/mrfc/provisioning/mrf/mrfc_MRPs.ldif
```

Note that the final file defines the IP address off the MRFP.

These files where copied from an existing MRFC running on a real TSP cabinet. However, some of the files needed modification before we could use them in our test configuration. In the schema file the application's distribution over the traffic processors is defined. For our test configuration we change it to only run on TP1, as this was our only Dicos traffic processor.

The MRFP we used was a prototype version and can be run on a native Linux system. We simply installed it on the Linux host and did not experience any problems. The MRFC utilizes a pool of MRFPs that it can distribute media sessions (connections) over. When a MRFP is booted it connects to the MRFC and announces that it is available as a resource.

The installation process was successful and the MRFC operated as expected. Thus we proceeded to the next step, which involved testing this system with the newly developed clients.

6.2 Teleconsulta Clients + Enablers + IMS

The Teleconsulta clients are based on the open source project Sip-Communicator [29]. When testing the clients and a stand alone Sip-Communicator very high latencies where experienced. Sometimes an INVITE could take as long as a minute to reach the other party, well after the caller had terminated their attempt to call. To make sure this was not due to the network or IMS core tests with CounterPath's X-Lite were conducted. When using the X-Lite clients the response times were only a few milliseconds.

When the source code for the Teleconsulta clients was examined one of the surprises was that many comments where written in Spanish. Although the development of these clients was done in Spain as a local project, the overall project is a collaboration with universities and an international

company, thus I expected that the comments would be in English. To facilitate future cooperation, I would suggest that all of the comments should be written in English.

Throughout the source code a number of small programming errors where found in the application. Each of these errors were reported to the application developers. Examples of these errors are:

- In the net.java.sip.communicator.impl.media the file CallSessionImpl.java on line 1631 has a reference to `Constants.H263_RTP_SDP` which does not exist. I had to make a workaround to compile.
- In net.java.sip.communicator.impl.gui.mainDoctor mainFrameDoctor.java there is a function named `public void añadirContacto(){` When imported into eclipse in Linux the ñ character is not found and there is a compile error. This could be fixed with a character set change, but it is a better way of working to stick to regular characters.

Some other issues with Sip-Communicator were also identified. Specifically, the SIMPLE presence implementation does not seem to be compatible with the PGM in IMS. Sip-Communicator can receive presence updates from other users, but when it tries to publish its own presence the result is an exception in the PGM:

```
2009-07-20 10:42:42,204 WARN  [pool-9-thread-2]
com.ericsson.pgm.presence.ps.business.PublicationDelegate -- Xml validation of
received publication content failed.
org.apache.xmlbeans.XmlException: Validation of XML failed. XmlErrors:
error: cvc-complex-type.2.4a: Expected elements
'person@urn:ietf:params:xml:ns:pidf:data-model
device@urn:ietf:params:xml:ns:pidf:data-model' instead of
'tuple@urn:ietf:params:xml:ns:pidf' here in element
presence@urn:ietf:params:xml:ns:pidf
```

If digest authentication is enabled in the CSCF, then the Sip-Communicator is unable to register and the request is rejected with a 403 User not authorized. This means that the HSS could not find the user. This probably occurs because the field username=<username> instead of digest username=<username>@<domain>. The correct operation can be demonstrated with X-Lite. With the X-Lite client the user registers, CSCF responds with 401 unauthorized including a nonce field prompting a digest response from the user's client. As a result the user's client re-registers with the correct headers and is authenticated. See the trace in Appendices A.3 and A.4. . Because the project is not using the latest build of Sip-Communicator it may be possible that this has been resolved in later releases.

6.3 ESSIM

ESSIM is built upon SuSE Linux Enterprise Server (SLES)10.2 from Novell. When using SLES right from the box and not pre-configured[‡] there is work to be done before the system is fully updated and working.

This configuration process is made more difficult because of some inconsistencies in the current documentation. As a result it can be very difficult for someone not experienced in Linux **and** VMware to follow the instructions. This is especially true since there a number of additional

[‡] HP manages Ericsson's internal computers, but did not manage the computers used in the laboratory for the testing in this thesis project.

packages that must be installed, but these are not mentioned in the current document. A list of these packages is shown in appendix A.1.

Another issue is that the virtual networks in VMware (which the VMs use to communicate) are host-only. No VIP has been implemented, thus the cluster is unreachable for all machines except for those on the same host. This is a major drawback because applications servers generally want to access external resources.

6.4 OMP DX2.0

Installation of the new DX2.0 was much improved over the cumbersome ESSIM installation. Given a virtual machine package with all the necessary TSP SAF sdp files and the MMAS already configured the installation was a breeze. The new single node configuration also makes it a lot faster and requires less performance from the underlying hardware. One drawback of this is that testing High Availability is not possible because the application cannot be distributed on several payloads when using the single node configuration. While there is still the possibility to run your own custom scripts and load any number of payload nodes this is very complicated.

Due to the single node configuration it was not possible to test the high-availability and redundancy functionality in DX. Instead we examined logging and fault management. Fault management in OMP concerns generating and terminating alarms that are sent to remote management stations. Logging in the MMAS uses Java standard logging, i.e., java.util.logging or log4j. If this was the only requirement for using logging, development would be very easy for developers. However, because the target environment is a multi-cluster system with many *instances* of MMAS there will be a lot of logging taking place. Therefore, logging is divided into two categories: (1) logging in TSP SAF and (2) traces which are Java logging of lesser importance. Tracing is used to reduce the amount of logging on the active controller node (CS). A minimal loglevel is defined and all Java logging requests below this value are considered tracing and is not stored as SAF logs on the CS. If the log request is above the defined value, then it is considered a SAF LOG request and it goes through Java CAF to be converted into a SAF LOG. There results of these SAF LOG requests are stored on the CS. The best scenario for developers (especially third party develpers) would be if the developer could simply enable logging and be finished. However, in order for Java CAF to successfully forward logs to SAF logging there are several settings needed to be made. By default all Java logging is considered tracing, so nothing is forwarded to SAF logging. To enable SAF logging the setting in the IMM (section 3.2.1.3) needs to be changed. By running the script `confd_cli` at the terminal, the IMM states can be changed.

```
MmasLogHandlerStreamConfig AppLogStream {
                          logHandler-streamName           AppLogStream;
                          logHandler-logFilePath          /MMASAppLogs;
                          logHandler-logFileName          AppLog;
                          logHandler-maxFileSize          100000000;
                          logHandler-highAvailabilityFlag 1;
                          logHandler-maxRecordSize        3500;
                          logHandler-actionWhenLogFull
SA_LOG_FILE_FULL_ACTION_ROTATE;
                          logHandler-numRotationFiles     10;
                          logHandler-recordFormat         "@Cr @Ch:@Cn:@Cs @CY-
@Cm-@Cd @Sv @Sl \"@Cb\"";
                            logHandler-logLevel            OFF;
}
```

By changing the logHandler-logLevel OFF to CONFIG, defined logging classes with a severity level above CONFIG will be SAF logged, while severities below will be considered tracing. This is done by the command: set Me 1 Common 1 Mmas 1 MmasLoggerConfigRoot 1 MmasLogHandlerRoot 1 MmasLogHandlerStreamConfig AppLogStream logHandler-logLevel CONFIG. The logHandler-logLevel setting should **not** be OFF by default, it is only in the virtualized environment that is set to OFF – unfortunately **no** documentation reflects this. The only way to know of this is to communicate directly with the developers.

When logging from a class the value this.getClass().getName(); is sent to the logger. If we want our logger to be handled in a special way, for example, with a different logger-level; we have to add it to the logger list. For instance , if this returns: ericsson.mmas.training.beans.MyJEEAppBean this needs to be appended to the IMM's logger-list, as shown below:

```
MmasLoggerConfigRoot 1 {
                logConfig-modelPollInterval 30000;
                global-logger-level         CONFIG;
                l4j-global-logger-level     INFO;
                MmasLoggerRoot 1 {
                    MmasLoggerGroup MMASLogging {
                        logGroup-name      MMAS_LOGGING;
                        group-logger-level "";
                        logger-list
com.ericsson.mmas.logging.AuditLogger,com.ericsson.mmas.logging.LevelManager,c
om.ericsson.mmas.logging.LoggingManager,com.ericsson.mmas.logging.LoggingConfi
g,com.ericsson.mmas.logging.LoggerLevels,com.ericsson.mmas.logging.RollingOutp
utStream,com.ericsson.mmas.logging.SafLoggingManager,com.ericsson.mmas.logging
.SafStreamConfig,ericsson.mmas.training.beans.MyJEEAppBean;
                    }

                }
```

When working with alarms the alarm log `fm.log` can be swamped with alarms about not having a connection to the licensing server (which it polls every 10 seconds), this because it is a virtualized platform without any licensing server, therefore this feature should be turned off. To lessen this problem the polling interval in the variable syncIntervalSeconds was increased from 10 seconds to 9999 seconds using the `confd_cli` manager, rather than turning off this alarm.

Alarms not defined in the IMM are stateless and therefore considered as alerts. Alerts are not show in the active alarm list. Whenever there is an alarm or a heartbeat an SNMP trap is sent to the management station. To determine if this really worked correctly and how it worked, a SNMP server was set up on the host computer. Because the virtualized OMP is on a host only network it was only possible to use the host machine as the SNMP server. As the SNMP server `snmptrapd` 5.4 [32] was used. With the command `snmptrapd -f -Lo disableAuthorization=yes 192.168.73.254:1162` the traps where logged at the console. Below is an example of a custom alarm as seen on the SNMP server:

```
2009-07-22 15:24:30 <UNKNOWN> [UDP: [192.168.73.1]:32780]:
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (31751) 0:05:17.51
SNMPv2-MIB::snmpTrapOID.0 = OID:
SNMPv2-SMI::enterprises.193.99991.1.4.5.1.5.0 = STRING:
"\"AlarmId=1,ExampleApplicationId=1,MeId=1\""
SNMPv2-SMI::enterprises.193.99991.1.4.5.1.4.0 = STRING: "Unknown alarm"
SNMPv2-SMI::enterprises.193.99991.1.4.3.0 = Gauge32: 2
SNMPv2-SMI::enterprises.193.99991.1.4.5.1.6.0 = INTEGER: 16385
SNMPv2-SMI::enterprises.193.99991.1.4.5.1.8.0 = INTEGER: 418
```

```
SNMPv2-SMI::enterprises.193.99991.1.2.1.0 = STRING: "The Germans are
coming!"
```

Sometimes when restarting the virtualized environment or during runtime the error "The component safComp=AlarmBridgeComp, safSu,sadNode=SC_2_1 failed due to : activeMonitorFailed(4)" appears and no alarms can be triggered. Apparently the stability of the environment is not 100% and the alarmBridge was never restarted.

In summary, the logging and fault management functionality in OMP works as intended and is possible to use but **only** if you know what you are doing. The documentation is incomplete and should be improved.

6.5 OMP DX2.0 + Enablers + Clients + IMS

In this step, the objective was to test all the components running together. Because DX is running in a host-only network, it is unable to communicate outside the host machine and we had to lower our ambitions. The reduced goal was to take two enablers (the group manager and the VirtualWaitingQueue) and add the logging and fault management functionality from OMP. These were then deployed in the DX environment and the logging and alarms were shown to work. However, an open question is what is the best way to explicitly implement the OMP functionality in the code of the enablers.

Originally the GroupManager and VirtualQueue enabler were deployed as `.war` files and are converged with the SIP modules. In order to be deployed on OMP from the DX Eclipse plugin the projects have to be packaged as Enterprise Applications (EARs). As the logging functionality is the same as in the original application, it does not need to be altered at all. The alarm functionality found in the test application used in section 6.4 was reused, by placing it in the EJB and EJBClient projects. The functionality for the alarms is implemented in Enterprise Java Beans (EJBs) with interface classes (i.e., as EJBClients). These are very nicely separated from the rest of the code in the OMPEJB project, making it easy to add this functionality to any other project, for instance the other enablers. If this is to be done, then the other enablers can simple be added to the EAR file, in order to make use of the OMPEJB. The resulting application is structured as shown in Figure 23.

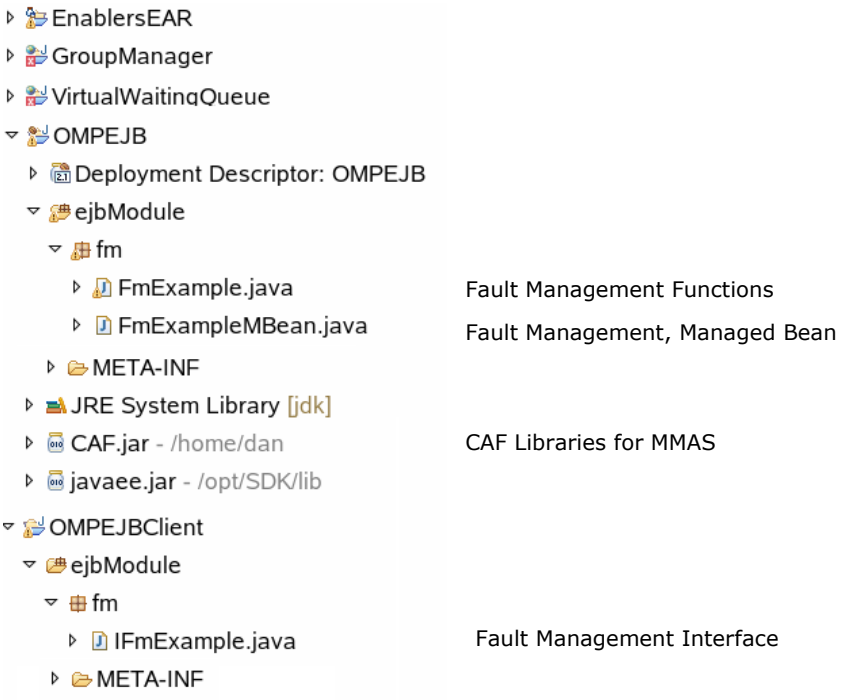

Figure 23: Application Structure

A question is when we want to trigger an alarm in the Group Manager. An issue is that we must be certain that we can know when a problem is fixed so we can clear the alarm. In the group manager there are several special exception cases, for example, the PGM exception and Enabler exception -- these can be used to judge when an alarm is appropriate. Only severe malfunctions in the application's operation should result in an alarm. Events that should raise an alarm include:

- Failure to connect to the PGM (i.e., no response from the PGM),
- Unable to read the ims.properties configuration file, or
- 401 Unauthorized and 403 Forbidden responses from the XCAP proxy.

All these events cause the Group Manager to completely malfunction. Minor errors that can either be handled as minor alarms or simply need to be logged are:

- The create group operation failed because the group already exists
- The delete group operation failed because the group does not exists
- Adding a member to this group failed, because the member already exists

All alarms are issued with a subtype, which provides a unique identification of the alarm in the TSP SAF Fault Management layer. This subtype is a string that might look like: com.ericsson.Groupmanager.xdms.connectionFailure. The initial part of this string is a vendorID ("com.ericsson.Groupmanager") that indicates which application is responsible for generating this alarm, the xdms part is the MajorID describing the component that is failing, and ""connectionFailure" is the MinorID describing in more detail what failed. In the sample application, all alarms where issued with the same subtype. Each alarm can have different severity levels and alarm messages. An example alarm might be: Critical GroupManager Alarm,

com.ericsson.Groupmanager.filestream.connectionFailure. String= Unable to read the ims.properties configuration file.

```
FmAlarmType Enabler_Error {
  majorType          193;
  minorType          104485226;
  moClasses          "[ Enablers ]";
  specificProblem    "Enabler, Cannot read ims.properties config";
  eventType          6;
  probableCause      418;
  severityLevels     "[ 5,4,0 ]";
  isStateful         True;
}
```

But during testing it was impossible to create new alarms through the confd_cli interface, the alarm can be created but when committing the changes it responds with a IMM: SA_AIS_ER_FAILED_OPERATION and the commit is rejected. The other option is to use a tool called SAFADE OaM Modeling Tool. This tool generates the IMM XML files that define the alarm settings, but then the user has to manually place them in the correct directories and this complicates the DX installation.

One problem with the enablers is that they rely on a configuration file `ims.properties` that is manually placed inside the domain/config directory. As OMP is a clustered environment it is not possible to manually insert configuration files in the domains. This has to be changed, a suggestion is to place the configuration information inside the EAR file. However, Configuration Management was outside the scope of these tests so the file was placed manually in the /config directory.

Overall the impression is that adopting an existing application to run on OMP was not as hard as originally thought. The logging was very easy, requiring no changes to the application, only changes in OMP configuration. However, fault management was a bit more tricky, but still very doable. The alarm managed beans could easily be reused for several applications in the same EAR project. Unfortunately, the hardest part remains: getting the application to run with high availability on several nodes with different amounts of redundancy. Fortunately, this was outside the scope of this thesis as the DX2.0 environment was configured for single-node use – however, the changes necessary for multiple nodes should be considered as future work (section 7.4.2).

7 Conclusions and future work

The aim of the thesis project was to investigate the environment for applications in a municipal setting. The investigation focused on the network layer as implemented in the form of IMS in the box (IITB) the application layer in the form of a videoconference application, and the use of OMP as a service platform. The overall conclusion is that there are a lot of opportunities for application and there exist a number of useful tools for developers of applications targeted at municipalities. We will consider these three foci in the following sections.

7.1 IMS in the Box

As a test environment and demonstration platform my experiences with IITB are very good. The system works exactly as IMS on TSP, but is scaled down and lacks the redundancy of a traditional IMS solution. It was very easy to add new components for running on the Dicos virtual machine, showing that the system functionality can be easily extended -- even by non-experts. However, some problems arise when multiple components are co-located on the same Dicos VM and Linux-host. For example, in the target system several components (HSS, CSCF, and MRFC) are **not** supposed to run on the same traffic processor. As a result there can be problems with components when using addresses on the virtual interfaces. Unfortunately, these problems are very hard to trace in the communication inside the traffic processor. If IITB is to be used as a real product for a customer, then more testing is needed and a greater separation of the sub-components is needed (see section 7.4.1).

The support responsibility for IITB needs to be cleared out. Are questions not regarding the installation supposed to go the individual components support groups? A conclusion from my work is that this can lead to unnecessary and time consuming communication between the different support groups, with each group forwarding the question to another group. I think that it would be better if the software packaging (SWP) team responsible for IITB took greater responsibility for supporting the complete solution. The software packaging team should be the first line of support; they can then refer the question to people in the individual component's organization if necessary. This issue has been brought up (internal to the company) and the resolution of this issue is assigned as a task to the SWP team. Another suggestion is that the SWP team should start an online forum similar to the OMP Support Forum where users can ask questions and can also read how other people have solved earlier problems. On a more practical note: the default PGM/XDMS settings should be reviewed and set to settings that are more useful from the start.

The SWP team has created some scripts that make it easier to provision users and associate them with service profiles. These scripts are very helpful for inexperienced users, but the SWP team should spend more time making IITB specific scripts or they should provide the users with component specific scripts and solutions. This is basically the same issue as raised earlier concerning support. At the moment the number of scripts is very limited, as only the most basic functionality is present. However, in this case (unlike that of the earlier support situation) I think the situation is best the way it is, i.e., that the SWP team does **not** create special solutions, but leaves it up to the users to communicate with the individual component teams.

Scalable IMS has a future in the Ericsson IMS portfolio. It can be used as a test system for IMS developers, as an evaluation platform for operators, or as a small scale solution for municipalities. It is useful to any organization in need of a complete IMS system, but with a limited user base and without the high availability required for a full-blown IMS solution.

7.2 Teleconsulta Application

The Teleconsulta enablers show that Ericsson's work with Sailfin and Glassfish provides a good platform for SIP enabled applications. The idea of creating a richer service layer is a good way to promote IMS. However, I do not think that Ericsson should create too many applications themselves. Instead, they should focus on promoting IMS as an enabler for applications, but leave the development of applications to third parties. The enablers seem to be very viable in a municipal network setting, with several additional applications that could potentially benefit from them (see future work section 7.4.3).

With regards to the application one strange aspect that I encountered was code comments in Spanish throughout the code and some Spanish function names. It would be better practice to write comments in English. Additionally, there is a lack of both system and product documentation.

The advantage of using Sip-Communicator as a code base is also debatable. Although, it is one of few open-source projects that support both SIP and video, it has several problems with regards to stability, authentication, and presence. If the application is going to be demonstrated to doctors and patients this part of the application needs to be improved quite a lot.

There are also some open issues regarding videoconferencing. Currently this seems to be unfinished and there is not (at the present time) a stable Ericsson MRFP product. At present only a prototype MRFP/Multicast unit was available for testing. If Ericsson is to offer a complete solution, then an MRFP product is necessary.

IMS in the Box has solved the problems with creating and demonstrating an application for potential users and customers. The platform contains everything that is necessary to use the application at an external location, thus avoiding the need to bring visitors to a special test environment at a company site for a demonstration. This greatly facilitates development with IMS as developers can create their own test environment, thus increasing the rate at which they can design, implement, and test their applications.

7.3 Open Multimedia Platform

During the thesis the OMP part caused the most difficulties. The DX2.0 environment was released well into the thesis project and the support group at Ericsson Multimedia was disbanded a few weeks into the thesis. Despite these challenges, the environment was set up and an enabler successfully installed and integrated with OMP/SAF functionality. A clear question is if these kinds of applications would benefit from additional work, in terms of integrating them with OMP. The work with Teleconsulta and effort to integrate the MRFC into IITB revealed some similarities between OMP and TSP and what kind of applications should run on them. From a technical perspective OMP might not be the right solution for municipalities, documentation can be hard to come by and the complexity of the system adds a lot of extra work for developers. The benefits in availability have to be carefully weighted against the drawbacks of a more complex system. In my thesis I promote IITB as a good small scale solution for a municipality core network, a solution using virtualization of TSP to reduce cost and without High-Availability. If we assume that the pricing between TSP and OMP is similar, it seems very strange to have a real OMP HA solution for service layer applications but not use the TSP for the core. So unless OMP could be made a lot cheaper than TSP (which is one if the goals with the platform), OMP is not the right solution for small-scale municipalities deployments. This does not mean that OMP is not the best option for big Operators that want the best availability and capacity.

With regards to the development environment, DX2.0 is a great improvement over the predecessor ESSIM. This is an important step forward, as there needs to be an easy to use toolbox for developers creating applications. However, there needs to be a lot more documentation about what is different between the virtualized environment and the real environment. This seems to be the same problem as with IITB. Once again the same support issue arises: Should questions be directed to the DX team or to the individual component teams? One thing that is better with DX2.0 than with IITB is the OMP support forum where users can ask questions and receive help in a common place. Third party developers also need a real environment to test their applications in. This should be provided, maintained and supported by Ericsson. One question that I asked myself early on was if SDS and the Eclipse plugin in DX2.0 could be combined. They are both targeted at service development for Ericsson equipment. From the beginning it seemed obvious that they could benefit from each other, but upon closer inspection I saw that SDS is IMS focused and more broadly targeted at users outside of Ericsson; while the DX2.0 environment is targeted only at OMP developers and OMP in turn is targeted at Ericsson Multimedia projects such as IPTV. However, the major reason not to combine the two tools is that to use the DX2.0 plugin you need to utilize the virtualized platform – but bundling in a virtualized environment that very few of the developers will need together with SDS will probably be extra work with little value. Hence my recommendation is **not** to bundle the two together at the present time. When OMP reaches more widespread use, this issue should be reconsidered; particularly if bundling the tools together could be beneficial for third party developers.

In 2009 Oracle bough Sun Microsystems, this may lead to drastic changes in SGCS and other software related to OMP. One of the major reasons for Open SAF in the first place was to avoid vendor lock in and to enable IT companies to control much of the platform stack. With Oracle acquiring SUN, Oracle controls a lot more of the platform. This is especially true as SUN can provide the hardware and SGCS; while Oracle can provide the databases needed for most applications. As a result there is a risk that Oracle could stop supporting SGCS and instead could create a very capable middleware and up solution, of course this might come at a much higher price to the end customer.

7.4 Future Work

7.4.1 Examine IITB performance

A clear future task is to determine where the bottlenecks are in the current setup. If IITB is to be used as a solution (i.e., a product) and not simply for development and demonstrations, then how should it be optimized? If all nodes are no longer to run on the same machine, then how should they be distributed for maximum performance? For example, the CSCF could run on one server with the HSS on a second server, the PGM on a third, and so forth. Should this be done by using VMware to virtualize Dicos. Some of the open questions are:

- What is the number of calls/second the CSCF can handle?
- What is the maximum number of simultaneous registered users?
- What is the maximum numbers of users that can be handled by a HSS?
- What is the maximum number of videoconference calls that can be handled by the MRFP?
- Is there a better default PGM/XDMS configuration?

A major (and highly important) task is to create a guide & package for installing the MRFC/MRFP into IITB for other developers. Additionally a number of other TSP applications could enrich the IITB solution, both as a demonstration platform and a customer solution.

7.4.2 OMP

The OMP platform offers a number of areas that require more investigation. The platform shows a lot of potential, but its full potential could not be shown in this thesis project. Some of the important questions are:

- What can be done to make OMP more appealing for third party application developers?
- Can all enablers be modified to run on OMP -- including high-availability?
- Can the enablers be run on a real target OMP system?
- Tests should be made of the performance and availability of the resulting system?
- Could the application run on TSP/Linux instead of OMP? What are the benefits and drawbacks of this approach?

7.4.3 Teleconsulta

An important activity for Teleconsulta is to find new applications that can use the enablers that have already been developed. Some applications include:

- VoIP call centers that can utilize the queue functionality;
- Business applications where users wants to share and discuss spreadsheets/documents; and
- E-learning applications where teachers and students can have videoconference and share course material.

Additionally, new general enablers to enrich the IMS service layer should be sought. A clear area of interest would be enablers for location-based applications.

An important decision is to consider an alternative software based (than the current SIP-Communicator) in order to enhance the user experience. The current level of performance of clients based upon this platform will not be acceptable to users.

References

[1] Nancy Ghoring and Stephen Lawson, "EarthLink to build out full Philadelphia Wi-Fi network", About.com, http://pcworld.about.com/od/wireless/EarthLink-to-build-out-full-Ph.htm date of last access: 2009.07.17

[2] Banhof, http://www.bahnhof.se/, date of last access 2009.07.17

[3] Zitius, http://www.zitius.com/www/live/start.aspx?TreeID=645, date of last access 2009.07.17

[4] STOKAB AB, http://www.stokab.se/ date of last access: 2009.04.10

[5] Emily Visser, "Broadband comes to Joburg", City of Johannesburg, March 2009, http://www.joburg.org.za/content/view/3554/266/

[6] Ericsson, Press Releases, http://www.ericsson.com/ericsson/press/releases/20080828-1246724.shtml

[7] JSR 281 Expert Group, JSR 281 IMS Services API, Java Community Process, " http://cds.sun.com/is-bin/INTERSHOP.enfinity/WFS/CDS-CDS_JCP-Site/en_US/-/USD/VerifyItem-Start/jsr281_api_fr_v1.0.pdf?BundledLineItemUUID=wmtIBe.lwsgAAAEhR2o_7Kt3&OrderID=OhhIBe.l8ToAAAEhKmo_7Kt3&ProductID=GC5IBe.pHtgAAAEb72ckexQ_&FileName=/jsr281_api_fr_v1.0.pdf"

[8] Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN), http://www.etsi.org/tispan/ date of last access: 2009.04.10

[9] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol", IETF, Network Working Group, RFC 3261, June 2002 http://www.ietf.org/rfc/rfc3261.txt

[10] M. Handley, V. Jacobson, and C. Perkins, "SDP: Session Description Protocol", IETF, Network Working Group, RFC 4566, July 2006 http://www.ietf.org/rfc/rfc4566.txt

[11] S. Kent and K. Seo, "Security Architecture for the Internet Protocol", IETF, Network Working Group, RFC 4301, December 2005 http://www.ietf.org/rfc/rfc4301.txt

[12] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, and J. Arkko, "Diameter Base Protocol", IETF, Network Working Group, DIAMETER protocol, RFC 3588, September 2003 http://www.ietf.org/rfc/rfc3588.txt

[13] Java Community process, JSR 319: Availability Management for Java, http://jcp.org/en/jsr/detail?id=319, date of last access: 2009.04.10

[14] VMWare Workstation, http://www.vmware.com/products/ws/ date of last access: 2009.04.10

[15] Wikipedia, http://en.wikipedia.org/wiki/Quiesce, date of last access: 2009.04.10

[16] C. Rigney, S. Willens, A. Rubens, and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", IETF, Network Working Group, RFC 2865, June 2000 http://www.ietf.org/rfc/rfc2865.txt

[17] Sun Microsystems Glassfish, https://glassfish.dev.java.net/, date of last access: 2009.04.10

[18] "99.999 or Five Nines Uptime", jiploo.com, http://www.jiploo.com/blog/99999-or-five-nines-uptime/ date of last access: 2009.09.02

[19] Sun Microsystems Enterprise Java beans, http://java.sun.com/products/ejb/, date of last access: 2009.04.10

[20] Open Source Community Sailfin, https://sailfin.dev.java.net/, date of last access: 2009.04.10

[21] Java Community process, JSR 289 SIP Servlet 1.1, http://jcp.org/en/jsr/detail?id=289, date of last access: 2009.04.10

[22] Service Availeability Forum, http://www.saforum.org/home, date of last access: 2009.04.10

[23] Open SAF, http://www.opensaf.org/Welcome-to-OpenSAF~151213~14944.htm, date of last access: 2009.04.10

[24] Christos Papazafeiropoulos, "Facilitating the adoption and use of the IP Multimedia System", Masters thesis, Royal Institute of Technology (KTH), School of Information and Communication Technology, March 2009 http://web.it.kth.se/~maguire/DEGREE-PROJECT-REPORTS/090317-Christos_Papazafeiropoulos-with-cover.pdf

[25] Veronica Kumlin, "Open Source SIP Application Servers For IMS Applications: A Survey", Master's Thesis, Linköping University, Department of Computer and Information Science, Linköping, Sweden, LITH-IDA-EX--07/066—SE, December 2007
http://liu.diva-portal.org/smash/get/diva2:17199/FULLTEXT01

[26] Open Multimedia Platform, Ericsson Whitepaper,
http://www.ericsson.com/ericsson/corpinfo/publications/review/2009_01/files/OpenMM.pdf

[27] Emil Erlandsson and Olle Ericsson, "Experience from Simulating TSP Clusters in the Simics Full System Simulator", Bleckinge Tekniska Högskola, http://www.bth.se/fou/cuppsats.nsf/all/97887c8645b9d16fc1256eae00799fb6/$file/thesis.pdf

[28] Göran Ahlforn and Erik örnulf, Ericsson's family of carrier-class technologies, Ericsson Review, Number 4, 2001, pages 190-95.
http://www.ericsson.com/ericsson/corpinfo/publications/review/2001_04/files/2001045.pdf

[29] Sip Communicator, http://sip-communicator.org/ date of last access: 2009.07.20

[30] Eclipse Integrated development environment, http://www.eclipse.org/, date of last access: 2009.04.10

[31] Ericsson Deveoper Connection, http://www.ericsson.com/developer/, date of last access: 2009.04.10

[32] Net-SNMP, http://www.net-snmp.org/docs/man/snmptrapd.html, 2009.04.10

[33] Sun Microsystems, Glassfish Developer Community, http://wiki.glassfish.java.net/Wiki.jsp?page=TipsAndBlogs, date of last access: 2009.04.10

[34] WMWare Fault Tolerance, http://www.vmware.com/products/fault-tolerance/, date of last access: 2009.04.10

[35] "Protecting Mission-Critical Workloads with VMware Fault Tolerance", Publisher Vmware, http://www.vmware.com/files/pdf/resources/ft_virtualization_wp.pdf

[36] X-Lite, http://www.counterpath.com/x-lite.html&active=4, date of last access: 2009.07.20

A. Appendix

A.1. Missing packages in SUSE 10.2 for ESSIM

- GCC standard Gnu Compiler
- VMware VIX
- VIX Perl library
- Expect: Expect is a tool for automating interactive applications

A.2. Missing packages in SUSE 10.2 for DX 2.0

- GCC standard Gnu compiler
- python-xml package

A.3. Wireshark Trace: Digest Authentication X-Lite

```
REGISTER sip:iitb.ericsson.se SIP/2.0
Via: SIP/2.0/UDP 192.168.0.105:2422;branch=z9hG4bK-d8754z-071aba48bd612251-1--
-d8754z-;rport Max-Forwards: 70
Contact: <sip:svanberg@192.168.0.105:2422;rinstance=f124046dfc1f55b3>
To: "svanberg"<sip:svanberg@iitb.ericsson.se>
From: "svanberg"<sip:svanberg@iitb.ericsson.se>;tag=ca3f8d05 Call-ID:
MTM4NzJkZDI2NjllMTc5YjliOTFmYTMwZDAwMDE4MjY.
CSeq: 1 REGISTER Expires: 3600 Allow: INVITE, ACK, CANCEL, OPTIONS, BYE,
REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO User-Agent: X-Lite release 1103d stamp
53117
Content-Length: 0

SIP/2.0 401 Unauthorized
To: "svanberg"<sip:svanberg@iitb.ericsson.se>;tag=1ab5b05906b0808fa49050a9a81c
From: "svanberg"<sip:svanberg@iitb.ericsson.se>;tag=ca3f8d05 Call-ID:
MTM4NzJkZDI2NjllMTc5YjliOTFmYTMwZDAwMDE4MjY.
CSeq: 1 REGISTER Content-Length: 0 Via: SIP/2.0/UDP
192.168.0.105:2422;branch=z9hG4bK-d8754z-071aba48bd612251-1---d8754z-;rport
WWW-Authenticate: Digest
realm="iitb.ericsson.se",domain="sip:authentication@iitb.ericsson.se",nonce="a
2f0a03d4c4695519f3fafe9050b05c7",stale=false,qop="auth",algorithm=MD5 P-
Charging-Vector: icid-value=1ab5b05906b0808fa4904f68b0ff P-Charging-Function-
Addresses: ccf="aaa://hss.ericsson.se"

REGISTER sip:iitb.ericsson.se SIP/2.0
Via: SIP/2.0/UDP 192.168.0.105:2422;branch=z9hG4bK-d8754z-d44bbf1efd208a6b-1--
-d8754z-;rport Max-Forwards: 70
Contact: <sip:svanberg@192.168.0.105:2422;rinstance=f124046dfc1f55b3>
To: "svanberg"<sip:svanberg@iitb.ericsson.se>
From: "svanberg"<sip:svanberg@iitb.ericsson.se>;tag=ca3f8d05 Call-ID:
MTM4NzJkZDI2NjllMTc5YjliOTFmYTMwZDAwMDE4MjY.
CSeq: 2 REGISTER Expires: 3600 Allow: INVITE, ACK, CANCEL, OPTIONS, BYE,
REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO User-Agent: X-Lite release 1103d stamp
53117
Authorization: Digest username=svanberg@iitb.ericsson.se
,realm="iitb.ericsson.se",nonce="a2f0a03d4c4695519f3fafe9050b05c7",uri="sip:ii
tb.ericsson.se",response="04b57d43b681f2dc8084bbe915d65d14",cnonce="12ebc03f85
37bee2292e552a1f1bc5ca",nc=00000001,qop=auth,algorithm=MD5
```

```
Content-Length: 0

SIP/2.0 200 OK
To: "svanberg"
<sip:svanberg@iitb.ericsson.se>;tag=1ab5b05906b0a08fa490524f0481
From: "svanberg"<sip:svanberg@iitb.ericsson.se>;tag=ca3f8d05 Call-ID:
MTM4NzJkZDI2NjllMTc5YjliOTFmYTMwZDAwMDE4MjY.
CSeq: 2 REGISTER Content-Length: 0 Via: SIP/2.0/UDP
192.168.0.105:2422;branch=z9hG4bK-d8754z-d44bbf1efd208a6b-1---d8754z-;rport
Contact:
<sip:svanberg@192.168.0.105:50444;rinstance=aeaff91e6b26a229>;expires=218
Contact:
<sip:svanberg@192.168.0.105:16888;rinstance=dfc77ed8ae2181fc>;expires=3195
Contact:<sip:svanberg@192.168.0.105:2422;rinstance=f124046dfc1f55b3>;expires=3
599 P-Associated-URI: <sip:svanberg@iitb.ericsson.se> Authentication-Info:
nextnonce="a2f0a03d4c4695519f3fafe9050b05c7",qop=auth,rspauth="6e20e12f28d5231
44e047d2ed977b31a",cnonce="12ebc03f8537bee2292e552a1f1bc5ca",nc=00000001 P-
Charging-Vector: icid-value=1ab5b05906b0a08fa4905205558f P-Charging-Function-
Addresses: ccf="aaa://hss.ericsson.se"
```

A.4. Wireshark Trace: Digest Authentication Sip-Communicator

```
REGISTER sip:iitb.ericsson.se SIP/2.0 Call-ID:
fa770b9cf2196fe06a5e0e4989df358f@0.0.0.0 CSeq: 5 REGISTER
From: <sip:hulken@iitb.ericsson.se>;tag=12db5eba
To: <sip:hulken@iitb.ericsson.se>
Via: SIP/2.0/UDP
192.168.0.105:5060;branch=z9hG4bK97a52cc86a14bcc2f066201a4a97a03b Max-
Forwards: 70
User-Agent: SIP Communicator 1.0-alpha3-0.build.by.SVN Windows Vista Expires:
3600 Contact: "hulken"
<sip:hulken@192.168.0.105:5060;transport=udp;registering_acc=iitb_ericsson_se>
;expires=3600 Content-Length: 0

SIP/2.0 401 Unauthorized
To: <sip:hulken@iitb.ericsson.se>;tag=b5428c9a06c0f08fa4d86b2559b0
From: <sip:hulken@iitb.ericsson.se>;tag=12db5eba Call-ID:
fa770b9cf2196fe06a5e0e4989df358f@0.0.0.0
CSeq: 5 REGISTER Content-Length: 0
Via: SIP/2.0/UDP
192.168.0.105:5060;branch=z9hG4bK97a52cc86a14bcc2f066201a4a97a03b
WWW-Authenticate: Digest
realm="iitb.ericsson.se",domain="sip:authentication@iitb.ericsson.se",nonce="7
40e9622cb2759b27cfbade935c92c08",stale=false,qop="auth",algorithm=MD5 P-
Charging-Vector: icid-value=b5428c9a06c0f08fa4d86ac17ce4

REGISTER sip:iitb.ericsson.se SIP/2.0 Call-ID:
fa770b9cf2196fe06a5e0e4989df358f@0.0.0.0 CSeq: 6 REGISTER
From: <sip:hulken@iitb.ericsson.se>;tag=12db5eba
To: <sip:hulken@iitb.ericsson.se> Max-Forwards: 70 User-Agent: SIP
Communicator 1.0-alpha3-0.build.by.SVN Windows Vista Expires: 3600
Contact: "hulken"
<sip:hulken@192.168.0.105:5060;transport=udp;registering_acc=iitb_ericsson_se>
;expires=3600
Via: SIP/2.0/UDP
192.168.0.105:5060;branch=z9hG4bKff962d0ec94c77b587916c0e61f609ee
```

```
Authorization: Digest
response="9537941feea6e814b1bcbb41361bb5cb",cnonce="xyz",username="hulken",nc=
00000001,qop=auth,nonce="740e9622cb2759b27cfbade935c92c08",realm="iitb.ericsso
n.se",uri="sip:iitb.ericsson.se",algorithm=MD5 Content-Length: 0

SIP/2.0 403 User Not Authenticated
To: <sip:hulken@iitb.ericsson.se>;tag=b5428c9a06c1108fa4d88ec676f4
From: <sip:hulken@iitb.ericsson.se>;tag=12db5eba Call-
ID:fa770b9cf2196fe06a5e0e4989df358f@0.0.0.0
CSeq: 6 REGISTER Content-Length: 0
Via: SIP/2.0/UDP
192.168.0.105:5060;branch=z9hG4bKff962d0ec94c77b587916c0e61f609ee P-Charging-
Vector: icid-value=b5428c9a06c1108fa4d88ea7aaf0
```

A.5. IITB Start Script

```
#!/bin/bash
function start {
  rcsybase start
  mcadmin setup
  rcmaia start
  rcjimslapd start
  rcmcinit start
  mcadmin loaddata

  #Configure EAS
  !/bin/bash
  . /etc/sysconfig/node
  . /etc/sysconfig/eascore
  /usr/bin/setServiceTrigger PGM 192.168.0.150 5160 # update with correct IP
  /usr/bin/addTrustedAS 192.168.0.150 #update with correct IP
  /usr/bin/addTrustedAS 192.168.73.1 # quickfix for enablers on the host

  /etc/init.d/./apache2 start

  #do once
  mkdir -p /opt/mirror/EAS/share/certificates
  cd /usr/java/jdk1.6.0_14/bin/
  ./keytool -genkey -dname "CN=,OU=,O=,L=,S=a,C=a" -alias tomcat -keyalg RSA
-storepass          mypass          -keypass          mypass          -keystore
/opt/mirror/EAS/share/certificates/easssl.keystore

  ecadmin install
  cpadmin install
  rcecinit start
  /usr/bin/setCscfAdminState 1
}

function stop {
  /etc/init.d/apache2 stop
  /etc/init.d/sailfin stop
  /etc/init.d/alexserver stop
  rcecinit stop
  cpadmin remove
  ecadmin remove
  rcmcinit stop
  rcjimslapd stop
```

```
  rcmaia stop
  mcadmin unsetup
}

if [ "$1" = "start" ]; then
  start
  exit
elif [ "$1" = "stop" ]; then
  stop
  exit

else
  echo bad input parameter, should be start or stop
  exit
fi
```

TRITA-ICT-EX-2009:103

www.kth.se