Open Source Development Tools for IMS Research (Invited Paper)

David Waiting
david@crg.ee.uct.ac.za

Richard Good
rgood@crg.ee.uct.ac.za

Richard Spiers
rspiers@crg.ee.uct.ac.za

Neco Ventura
neco@crg.ee.uct.ac.za

Department of Electrical Engineering
University of Cape Town
Rondebosch, South Africa

ABSTRACT

The 3GPP IMS is a next generation network architecture aimed at bringing the features and rich services of the Internet to the telephony world. Traditionally telephony products are developed by large companies with access to the proprietary solutions required for PSTN products. However, the shift to a packet-switched architecture and open Internet protocols has increased the developer base to include the huge community of web-developers.

Consequently there are currently several open source software projects that aim to provide proof-of-concept implementations and research tools for promoting the development and adoption of IMS technologies. This work investigates the tools created by four open source IMS projects and incorporates these tools into a practical IMS test-bed framework. Evaluations are performed that demonstrate the capabilities and limitations of these tools in providing rich services to IMS users.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Packet-switching networks*

Keywords

IMS, open source, testbed

1. INTRODUCTION

Since the invention of the telephone in the late 1800s there have been few changes in the voice telephony world. Network operators have thus far shown little inclination to evolve from the status quo because they have invested huge amounts in their current PSTN infrastructure and up until now have reaped huge profits. Consumers, however, are becoming increasingly frustrated by the limited services and exorbitant costs of traditional telephony and have embraced technologies such as Voice over IP (VoIP), Instant Messaging (IM) and video chat, which are all freely available on the Internet. Web-based companies such as Skype and Google are eroding voice revenues and turning network providers into dumb bit pipes.

The 3GPP, having recognised the benefits of an Internet-like architecture, are in the process of developing a new packet switched communications framework known as the IP Multimedia Subsystem (IMS) that is accessible from several different connectivity access networks, including UMTS, WLAN and DSL. In developing the IMS specifications the 3GPP have looked to the Internet Engineering Task Force (IETF) to provide the protocols that will enable a new service-oriented network based on the principles of the widely popular Internet. Other standards bodies, such as TISPAN and 3GPP2, have recognised the benefits of this Internet-like architecture and have adopted the 3GPP IMS into their own specifications.

Protocols designed for the Internet, including SIP, Diameter and HTTP, are used extensively in the IMS. An interesting side-effect of this switch to Internet protocols is that the telephony development realm, once limited to experts in SS7 and other PSTN protocols, is now open to a multitude of web developers. The benefits of a large development base have been proven in the Internet. The phenomenon commonly known as Web 2.0 has resulted in a new age of community-based websites with millions of users, often not developed by large corporations, but rather a few college students with an interest in web technologies. This trend has propagated to the world of IMS where developers worldwide have begun experimenting with new open source applications and releasing them free of charge to the Internet community.

The aim of this work is to identify, investigate and evaluate four open source tools currently available for IMS research. These tools span various elements of the IMS architecture including the user equipment, core network and application server. In order to make accurate qualitative evaluations of the tools they are incorporated into a practical test-bed framework where they can be subjected to realistic use case scenarios. This paper focuses primarily on the open-source projects created as part of the University of Cape Town's

IMS research initiative. The projects include a client emulation tool, a QoS policy control framework, an IPtv video streaming server and a distributed video conferencing application.

2. RELATED WORK

The IMS Communicator [9] is an open source IMS Client emulator built on the JAIN SIP Reference Implementation (RI) and the Java Media Framework API by the researchers at PT Inovação Portugal. The IMS communicator extends the SIP communicator project with new IMS-specific features, such as the implementation of the AKAv1 authentication algorithm, use of the IMS Public User Identity (IMPI) during registration, subscription to the reg event package, support for the precondition mechanism and early media capabilities. In order to implement these features the project contributed to the extension of the JAIN SIP RI that previously did not support the 3GPP extensions to SIP.

The Open Source IMS Core project [8] by the Fraunhofer Institute FOKUS is an implementation of the three IMS core network Call Session Control Functions (CSCFs) and the Home Subscriber Server (HSS). The CSCFs are built on top of the well-known SIP Express Router (SER) [10] maintained by Iptel.org, which is renowned for its reliability and performance under high-load conditions. The HSS on the other hand is implemented in JAVA and provides a web user interface for the rapid provisioning of users, application servers and initial filter criteria. The Open Source IMS Core is a remarkably stable and reliable standards compliant IMS reference implementation and hence is chosen to provide the core network of the testbed framework described in this paper.

3. UCT IMS CLIENT

The first IMS tool evaluated in this paper is a well-known client emulation tool, the UCT IMS Client [11]. Made public in November of 2006, it was the first IMS client released to the open source community. The aim of the project is to provide true IMS signalling, proof-of-concept implementation of several rich services, and a mechanism with which to test other IMS network components. The aim of the project is not to create a stable consumer grade product but rather a platform from which new IMS enablers can be implemented, evaluated and refined.

The implementation of the client is achieved by the use of several free open source libraries: the oSIP and eXosip libraries for SIP signalling; the gstreamer framework for media coding, decoding and transport; the libcurl library for HTTP support; the libxml library for XML parsing; and the GTK library for the graphical user interface. The client itself is released under the GNU Public License version 3 (GPLv3) that allows users the freedom to modify and redistribute the software for their own purposes. This is particularly useful for research projects that aim to produce innovative services that require modifications to the client software, some of which are discussed in this paper. We now evaluate the features of the UCT IMS Client that have allowed for the evaluation of several interesting performance metrics.

3.1 IMS-Level Registration

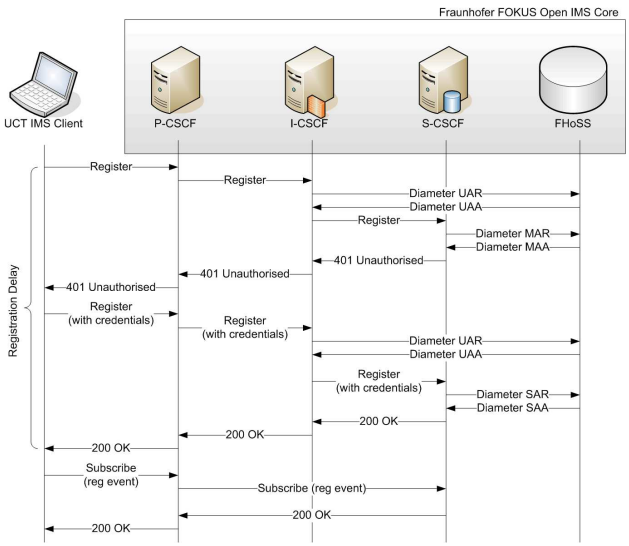

Figure 1: IMS-level registration flow.

The first evaluation measures the time required for IMS-level registration with the core network over several different access network technologies. The IMS-level registration flow is illustrated in Fig. 1. The User Equipment (UE) sends an initial REGISTER request to the P-CSCF address that is manually configured into the client. This requires that the UE perform a DNS look-up on the P-CSCF address before sending the request. The P-CSCF then forward the request to the I-CSCF that, with the help of the HSS, relays the request to a suitable S-CSCF. In order to authenticate the user the S-CSCF challenges the UE with a 401 Unauthorised response, which contains a nonce value. The client inputs the nonce value and the user credentials into the AKA algorithm and generates a reply that is inserted into a second REGISTER request. On receiving the request with a correct authorisation header the S-CSCF registers the user's Public User Identity (IMPU) and associates it to the client's IP address. The S-CSCF replies with a 200 OK message in order to inform the UE that it is successfully registered on the network. This message also serves to inform the UE of any other IMPUs that are available to be registered by the user and the *Service-Route* informing the UE which route should be followed by all subsequent requests.

The client and the P-CSCF then both subscribe themselves to the user's *reg* event, so that they will be notified if the user's registration state changes for some reason. The subscriptions to the *reg* event do not form part of the registration delay, nor does the PDP context activation for the wireless networks. These are therefore omitted from the measurements, however, the DNS look-ups do form part of the registration delay. Consequently, the complete registration delay is measured from the first DNS look-up on the P-CSCF to the time that the 200 OK response for the REGISTER request is received. The UCT IMS Client features a registration delay timer that is used to conduct these experiments.

The testbed framework includes terminals connected to three different access network technologies: Fast Ethernet LAN,

Table 1: IMS Registration delay results.

	LAN	HSDPA	EDGE
Minimum	0.21	1.13	7.05
Mean	0.31	1.57	9.39
95th Percentile	0.35	1.91	16.91
Std Dev	0.04	0.30	3.05

HSDPA and EDGE. The LAN connection is 100 Mbps duplex and the UE is connected directly to the P-CSCF. The HSDPA connection offers a theoretical maximum down-link access speed of 1.8 Mbps. The EDGE connection on the other hand can only support a maximum down-link speed of 236 kbps. In the case of the wireless access technologies the IMS core network is located within the UMTS core network. This is to prevent IP routing delays through the public Internet and therefore offers a realistic emulation of a carrier-grade IMS implementation.

For the purposes of the experiment the UCT IMS Client registers 100 times over each different access network with the Open IMS Core. Table 1 shows the results of the experiments measured in seconds.

3.2 IMS Call Setup Delay

The second evaluation measures IMS call setup delay. Typically, Internet SIP calls require relatively little signalling as there are no QoS requirements and reliability is not a crucial issue. The IMS, however, strives to offer a quality of experience equal to if not better than traditional circuit-switched telephony [6]. Therefore, the 3GPP have stipulated a strict call setup procedure [4] that ensures both the originating and terminating access networks have provisioned adequate channels in order to ensure minimal delay and packet loss, which would negatively impact the media quality.

An Internet SIP call starts with the User Agent Client (UAC) sending an INVITE request. This request receives one or more provisional responses from the User Agent Server (UAS) followed by a 200 OK final response. The UAC then sends an ACK request after which the clients begin to stream the media component of the call.

The call setup procedure for an IMS call is slightly more complex as the SIP precondition and reliable provisional response mechanisms are used. This results in significantly more signalling. In fact where a successful SIP call routed through the IMS architecture would only require 23 SIP messages, an IMS call requires a minimum of 59 messages. The difference between the call setup procedures is illustrated in Fig. 2. For simplicity the signalling through the core network elements is not shown.

The IMS call setup starts with an initial INVITE request sent from the UAC to the P-CSCF containing a media offer. The request is relayed to the UAS through the originating IMS core network and possibly through a terminating network if the two clients are located on different networks. The UAS replies with a 183 Session Progress response with a media response. In SIP provisional responses are usually not sent reliably, but since the provisional response is essential to the IMS call setup procedure it requires acknowledgement

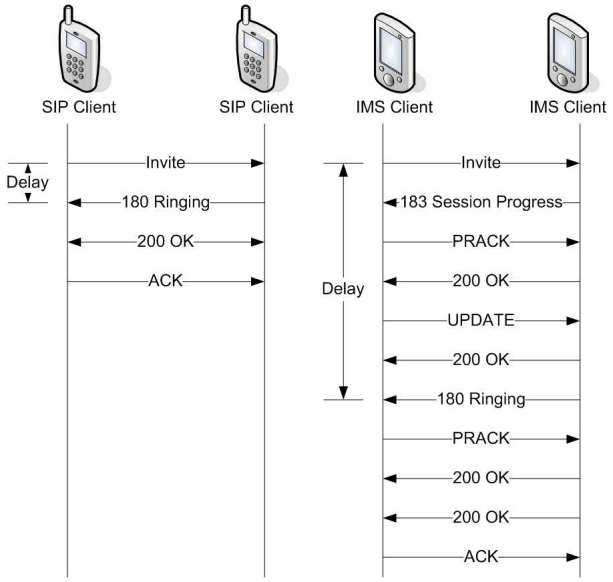

Figure 2: Comparison between SIP and IMS call setup procedure.

Table 2: SIP call setup delay results.

	LAN	HSDPA	EDGE
Minimum	0.42	1.12	6.06
Mean	0.49	1.71	11.34
95th Percentile	0.55	2.29	21.84
Std Dev	0.06	0.33	7.21

in the form of a PRACK message. Once the originating network has provisioned suitable resources for the call the UAC sends an Update message informing the UAS that it is ready to start a call. If the terminating network has also provisioned sufficient resources for the call the UAS replies with a 180 Ringing response and informs the callee that there is an incoming call. Again this provisional response elicits a PRACK from the UAC. If the callee accepts the call then the UAS sends a 200 OK message followed by the UAC sending an ACK. The time taken for the callee to answer the call is an uncontrollable variable and therefore does not form part of the call setup delay.

Measurements for the experiment are performed by the UCT IMS Client call setup delay timer. The call setup delay is measured from the first DNS look-up until the reception of the 180 Ringing corresponding for the initial INVITE transaction. The calls contain both an audio and video component. As with the registration delay the experiments are performed over three access networks and 100 measurements are taken for each network. Table 2 shows the call setup delays measured in seconds for Internet SIP calls and Table 3 shows the delays measured for IMS calls.

3.3 Discussion

The UCT IMS Client has demonstrated the suitability of the different connectivity access network technologies for IMS services. The evaluations find that the both the LAN and HSDPA access networks provide adequate mean registration

Table 3: IMS call setup delay results.

	LAN	HSDPA	EDGE
Minimum	0.82	3.21	20.27
Mean	1.05	3.95	27.56
95th Percentile	1.27	5.25	45.87
Std Dev	0.23	0.75	12.46

delays of 0.31 and 1.57 seconds respectively. On the other hand, the EDGE network has a mean registration delay of almost 10 seconds. However, as an IMS client is only expected to register with the network on start-up and periodically refresh its registration state this delay is not critical.

The evaluations further find that the increased signalling of an IMS call greatly prolongs the call setup delay. The mean HSDPA delay of 3.95 seconds is not ideal but still acceptable. However, a mean delay of 27.57 seconds and 95th percentile delay of 45.87 seconds clearly indicates that slow access networks such as EDGE provide unacceptable session setup delays. Much of this delay stems from the SIP retransmission timeouts that were originally designed for the Internet environment and are therefore easily triggered in wireless access networks with large round-trip delays. A possible solution is to adjust the SIP retransmission timeouts depending on the current access network [15].

4. UCT IPTV SERVER

The UCT IPtv server is a SIP application server that streams up to three channels to multiple destinations. The server is built on top of the eXosip library for SIP signalling and the gstreamer library for media delivery and it is released free on the Internet under the GPLv3 licence. The server is compatible with any client that is capable of receiving and decoding the H.263-1998 video standard and MPEG1 audio standard. The server should not be confused with a video-on-demand server that serves a different media stream to each client. Its primary goal is to serve a limited number of packet-based media streams to as many clients as possible similar to regular digital terrestrial and satellite television broadcasts.

The server is designed to fit tightly within the IMS architecture therefore unlike other IPtv solutions it uses SIP exclusively for signalling. The benefit of this design is that the pre-existing IMS QoS, service provisioning and charging mechanisms can all be reused for this service, hence leveraging the service delivery platform that IMS provides. Furthermore, this reduces the complexity of client software as the existing SIP stack can be reused.

The IPtv server is provisioned in the HSS as an application server. This enables the network operator to add and remove users from using this service by adjusting their initial filter criteria. Users that have the IPtv service enabled in their user profile are able to join a channel by submitting an INVITE request to the application server that includes their preferred media IP address and port numbers for both the audio and video components of the video stream. The IPtv server examines the request URI and adds the user to the appropriate channel. The server then treats the call as a normal IMS call, except that in its response it does not supply any media ports as it does not expect to receive media from the client. In order to end the IPtv session the UE sends a SIP BYE request. The IPtv server then removes the client from the channel as illustrated in Fig. 3.

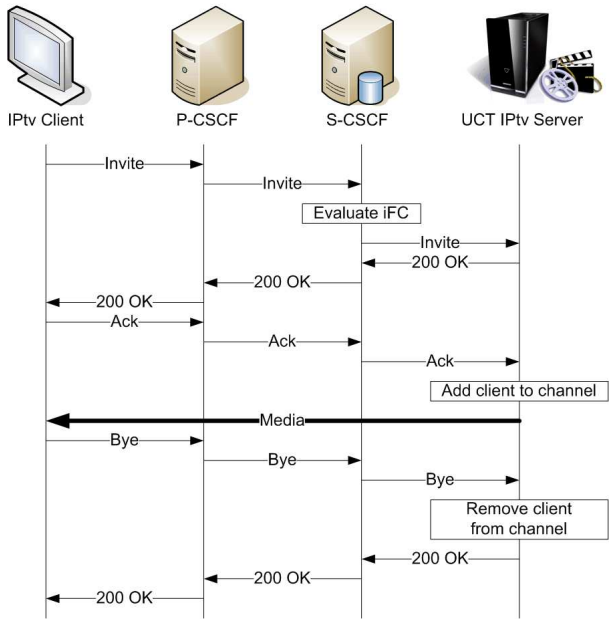

Figure 3: Joining and leaving an IPtv session.

The server supports both unicast and multicast IP addresses. For smaller network environments the clients can use unicast addresses but in larger networks it is expected that the clients should use multicast addresses, thus reducing the network load on the server.

4.1 Access Network Capabilities

The server is able to stream video at several different qualities depending on the usage requirements. For example if the server is to be used to supply small handheld terminals then a very low media quality can be specified. On the other hand if the server is supplying home theatre systems then a very high video quality is used. This video quality must be specified when the server is initialised and does not change on a per-user basis.

In this section we evaluate reasonable video quality settings for the two access network technologies, LAN and HSDPA. For each access network type the server streams three qualities IPtv to the client - low, medium and high, where low is 100,000 bps average bit rate, medium 500,000 bps and high 900,000 bps. On the client side the total traffic throughput is measured as well as the packet loss. For the purposes of the evaluation a one minute video clip was streamed from beginning to end and the resultant video feed at the client side was captured using the open source Wireshark Network Protocol Analyser. The UCT IMS Client is used to setup and receive the IPtv session.

Theses metrics are directly linked to the quality of video received by the client. A low traffic throughput indicates a poor quality media stream whereas high packet loss usually

Table 4: Traffic throughput (Mbit/s) / packet losses (%) at user equipment.

	LAN	HSDPA
Low Quality	0.351 / 0.00	0.350 / 0.06
Medium Quality	0.554 / 0.13	0.554 / 0.41
High Quality	0.964 / 0.07	0.965 / 0.19

Table 5: Server traffic generation (Mbit/s).

No. of Clients	1	2	4	8
Low Quality	0.35	0.70	1.40	2.79
Medium Quality	0.56	1.11	2.22	4.43
High Quality	0.97	1.93	3.86	7.67

results in lost video frames and audio disruptions. Therefore, the administrator should choose a video quality setting that provides a good compromise between traffic throughput and packet loss.

4.2 Server Traffic Statistics

In some implementation scenarios the clients will not be using multicast addresses or the server may offer many channels. In both these situations the server is required to stream several simultaneous media streams requiring a great deal of network bandwidth.

In this evaluation we examine how the bandwidth requirements increase according to the number of unique video streams being served and the quality of the IPtv sessions. The same three video quality settings used in the previous evaluation are streamed to a varying number of clients. The results of this evaluation are shown in Table 5.

4.3 Server Resource Usage

The UCT IPtv server is responsible for trans-coding the video and audio streams into the H263 and MPEG1 codecs respectively. This along with the streaming of several media streams places strain on the resources of the machine hosting the server. If the server becomes overloaded then there is the possibility that the video will not be encoded correctly and will result in a poor viewing experience for the clients.

In this evaluation the IPtv server was run with one, two and three channels, each at three different quality settings. The evaluations are performed on an Intel Pentium Dual Core 3 GHz machine equipped with 1 GB of RAM. The *pidstat* utility was used to record the CPU and memory usage of the application. A sample was taken every second for three minutes and the results averaged. The results of the evaluation are shown in Table 6.

Table 6: Mean IPtv server CPU / Memory usage (%).

No. of Channels	1	2	3
Low Quality	28.6 / 2.25	52.87 / 4.33	90.38 / 5.45
Medium Quality	32.24 / 2.61	61.86 / 4.28	92.40 / 5.75
High Quality	33.27 / 2.27	62.29 / 4.06	96.76 / 6.13

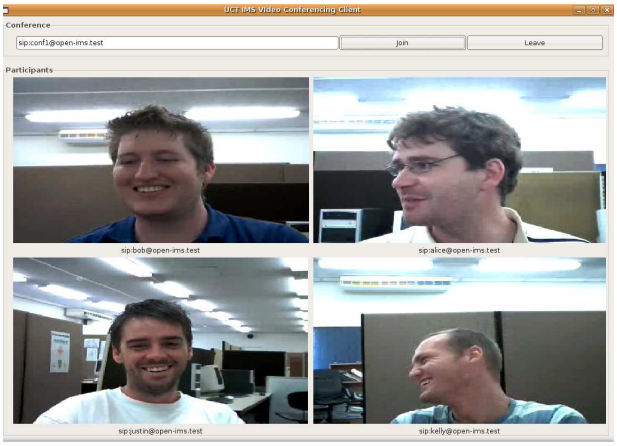

Figure 4: A screen-shot of the UCT IMS Conferencing client in a four-way video call.

4.4 Discussion

The first evaluation demonstrates that the HSDPA access network provides comparable performance to the LAN network for all the video quality settings tested, with packet loss at less than a percent for both networks at the highest video quality. The second evaluation shows that a linear increase in server bandwidth is required to support new clients. Furthermore, the bandwidth requirements are greatly increased depending on the quality of video being served.

The final evaluation shows that the CPU and memory resource requirements of the IPtv server are more influenced by the number of channels being served rather than the quality of video. This is due to the fact that the server requires a large amount of resources in order to trans-code the video sources into the media codecs that can be digested by the clients. In a commercial environment it is expected that the trans-coding of the video should be performed ahead of time in order to reduce the run-time resource requirements of the IPtv server.

5. UCT VIDEO CONFERENCING FRAMEWORK

The IMS provides a platform for the creation of several complex services that reuse the core network signalling, charging and QoS capabilities. One such service is teleconferencing - however, at present much research is still required in order to determine the optimal conferencing network architecture. The IMS specifies that signalling and media traffic are treated separately by the network elements and are routed on different IP paths. Thus at the core of the problem is how to manage the two traffic types between a large number of nodes. Every conference participant must send and receive media from every other participant thus creating a large amount of signalling and media traffic in order to setup even a small conference.

5.1 Architecture

There are three options for the media flow. The media can be controlled in a distributed fashion whereby every participant sets up a duplex stream with every other participant's unicast IP address. Alternatively the participants can utilise

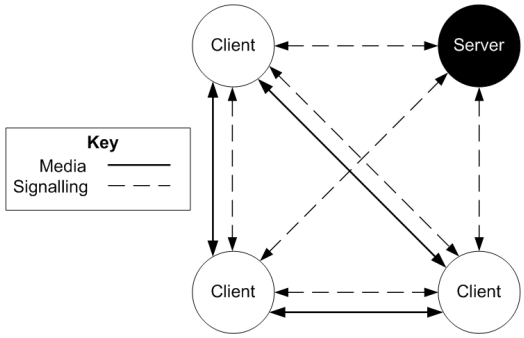

Figure 5: The UCT Conference Framework implements a distributed media and hybrid signalling scheme.

multicast IP addresses that allow a single IP stream from a client to be forked to several different participants, hence conserving the upload traffic. The last option is to have a centralised media server that multiplexes the incoming media from each conference participant and transmits the multiplexed media stream to all participants.

Unfortunately multicast is not extensively implemented in Internet IP routers and can therefore only be used in certain closed environments. In order to conserve bandwidth resources the centralised media server is an attractive option. However, this requires a dedicated media server with ample resources as even a small conference produces a great deal of media traffic. Therefore, in order to avoid the complexities and resource costs of implementing a centralised media server the UCT conferencing architecture implements a distributed media model.

As with the media there are several options for the signalling traffic. The signalling can be distributed amongst the nodes so that all participants control the signalling to each other. A second option is to have a focused signalling point that handles all signalling requests.

The distributed signalling approach removes the requirement for a dedicated conference coordination server. Unfortunately this introduces the requirement that all participants have knowledge of the other participants addresses before the conference starts, as there is no way for participants to automatically find out this information. The focused signalling approach solves this problem as the server can keep a record of the current conference participants and inform current participants of any joins or leaves. It is beneficial for each participant to negotiate each session with the other participants individually so that the network can assign resources, and the two clients can use the media types and codecs that they both support. However, with focused signalling this is very hard to achieve without a great deal of back-and-forth messaging and consequently, a high signalling overhead. For this reason the UCT conferencing framework implements a hybrid signalling scheme in which a conference participation is focused to a central server but individual session setup is performed in a distributed manner.

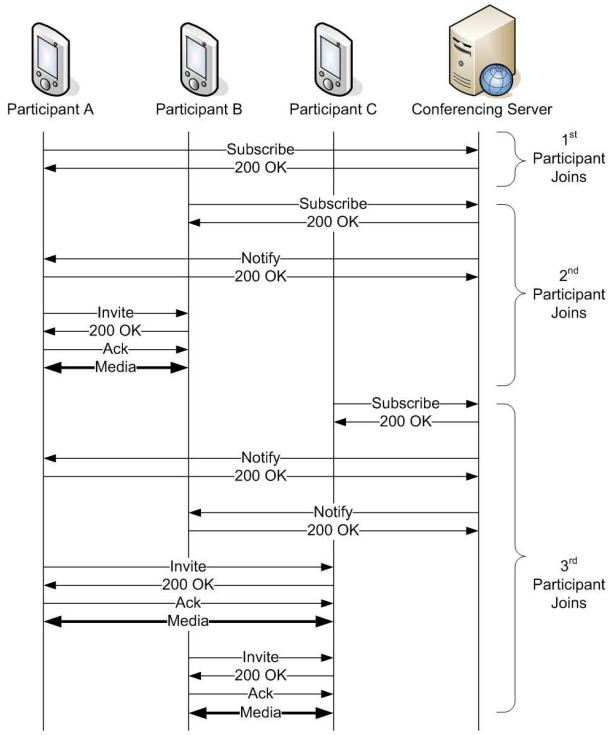

Figure 6: Conference signalling between three participants and the conference coordination server.

5.2 Signalling

The conference coordination server implements the SIP conference event package in order to maintain a list of conference members. When a user wants to join a particular conference they must send a SIP SUBSCRIBE request to the conference server. The conference server adds the user to the current list of participants. It then generates SIP NOTIFY messages to all existing conference members containing the XML-formatted SIP URI of the new user in the body of the message.

On receiving the NOTIFY request, the existing conference participants each send a SIP INVITE message to the new member. The new member processes each INVITE request separately, negotiating appropriate media port numbers and codecs. This also gives the access network opportunity to allocate appropriate resources and to bill each session separately. The new participant replies to each new INVITE request with a 200 OK response following which media can flow between the user terminals.

This architecture ensures that clients with lesser capabilities can still join the conference without reducing the experience for the other participants. For example, a client that does not support video can still join the conference with voice only without affecting the video streams between the other members.

5.3 Signalling Overheads

The hybrid signalling scheme utilised by the UCT conferencing framework is based entirely on SIP. While the IMS network operator benefits from this as it can now easily con-

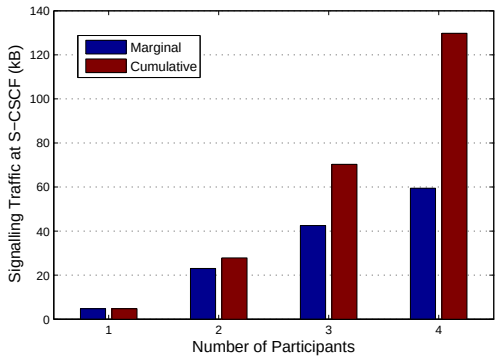

Figure 7: The measured SIP signalling overhead at the S-CSCF when a new participant joins the conference.

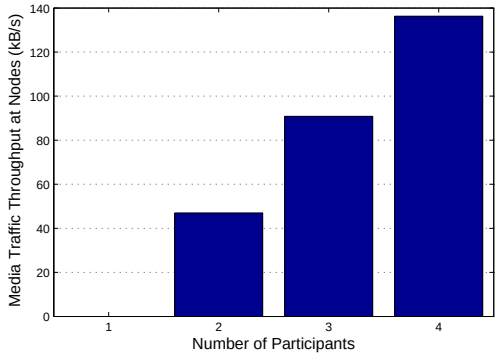

Figure 8: The measured media traffic at the conference client nodes.

trol and bill for the individual sessions this does also place extra overheads on the IMS core network.

We now evaluate the signalling overheads associated with setting up a conference. The S-CSCF routes all messages in the IMS architecture, therefore signalling overhead is measured at this node. Wireshark Network Protocol Analyser is used to record all measurements. Fig. 7 shows the marginal and cumulative signalling overhead when clients join the conference. The traffic includes both the conference event subscriptions and subsequent notifications to and from the conference coordination server as well as the signalling between the individual conference participants. Note that the marginal signalling overhead increases as each new participant joins the conference as the current participants must all individually invite the new member.

5.4 Traffic Overheads

The distributed media flows create a large amount of media at each node. In this evaluation we measure the total duplex media throughput on each client node for varying numbers of conference participants. The data was collected by Wireshark measured over a 60-second period. The media traffic at all nodes is very similar as each node has an equal number of incoming and outgoing streams. The mean media throughput of all nodes in the evaluation is illustrated in Fig. 8.

5.5 Discussion

The results of the evaluations show that the marginal signalling overheads increase with every new participant that joins the conference. This causes an exponential increase in signalling load with every new conference member. While the media scales linearly with every new conference participant it still places a large bandwidth burden on the conferencing client nodes. Thus it can be concluded that although the framework offers several advantages in terms of simplicity and flexibility it is only suitable for conference sessions with a small number of participants. A centralised media and focused signalling architecture is required for larger conference sessions.

6. UCT POLICY CONTROL FRAMEWORK

Policy frameworks allow a network operator to have strategic control over every aspect of the network architecture ranging from resource reservation to end user experience, using pre-defined policies. While policy control already exists at various layers in existing networks the policies are generally static and have little interaction with applications. As networks converge and the number of services increase and they become more personalised, end to end policy co-ordination will become critical.

Standards bodies have recognised the benefits of such a management framework to control resources and admissions in the transport layer; most notably the 3GPP have defined the Policy Control and Charging Architecture (PCC) [1] and ETSI TISPAN have defined the Resource and Admission Control Subsystem [7]. These architectures specify only logical architectures and not physical implementations; furthermore the systems are highly complex and will require significant investment to deploy in a commercial environment. Operators will find it difficult to commit the necessary capital unless the systems have been tried and tested.

The UCT Policy Control Framework is a 3GPP compliant, open source and freely distributed architecture specifically designed for QoS Policy Control in a single IMS domain [11] - the framework works in co-ordination with the FOKUS Open Source IMS Core. It is largely based on the 3GPP PCC specification and defines a Policy Decision Function (PDF), Policy Enforcement Point (PEP), Policy Repository and Web Management Interface. The framework provides a flexible environment for the easy creation and execution of policies encouraging further research and innovation in the field.

6.1 Policy Definition

With the rapid creation of multimedia services envisaged in the IMS environment the network dynamics will be constantly changing. Consequently the framework that controls resources and admissions needs to be flexible and adaptable - specifically network operators need to be able to rapidly create and add policies that govern the call admission and policy rule creation processes.

The UCT Policy Control Framework uses the eXtensible Markup Language (XML) to represent policies; this format

has been adopted as the method for storing policy information in numerous IMS services. The Open Mobile Alliance (OMA) have standardised an XML Document Management (XDM) element [12], this element allows the efficient storage and retrieval of XML documents from a centralised server. The XML Configuration Access Protocol (XCAP) has been defined to transmit the documents [13]; this protocol runs over an HTTP connection and the signalling takes place outside of the IMS Core.

We adapt the format specified in RFC 3644 - Policy Quality of Service Information Model, in order to define the policy structure [14]. Essentially this states that each policy is made up of rules that define actions and conditions - if a condition is met an action is taken. OpenXCAP is an open source implementation of the XDM specification that is used for centralised policy storage [5], any policy type can be stored as long as the defined structural rules are adhered to. Additionally each PDF stores localised policies; these policies are assigned a higher priority and may override the global policies stored in the XDM Server. In this way a hierarchical system of policy control is created whereby the centralised policies control resources and admissions throughout the entire network, while each PDF is responsible for its own domain.

6.2 Policy Processing

The UCT PDF is a sub-element of the Policy and Charging Rules Function that administers policies to control resources and admissions; the architecture does not yet include any charging functionality.

The element is Java based and implements the Diameter Rx interface to an Application Function in the control layer [3]. This interface is used to pass authorisation requests to the PDF, and to pass session requests to the control layer. A previously undefined interface is implemented between the PDF and the XDM Policy Server that uses XCAP to retrieve and store policies. This interface could utilise Diameter for more secure communication, but this would compromise the flexibility of the system because each policy format would require a new Diameter Application.

The UCT PDF extracts session information from Authorisation Requests and performs admission control and creates dynamic policy rules for each session. These processes are carried out by a Decision Engine (DE) that is governed by policies. The DE contains Policy Processor Blocks; these modular classes are specific to a policy type and allow for the easy addition of new policies. This adaptable system allows network operators to flexibly control resources and admissions by rapidly adding and removing policies like building blocks.

Simple domain policies are included with the freely distributed UCT Policy Control Framework by default; these policies allow a network operator to define certain network constraints including authorised domains, QoS classes and codecs. It is left up to the network operator to develop more complex and integrated policies.

For each session request, policy rules are created that define the session and how they must be treated in the transport

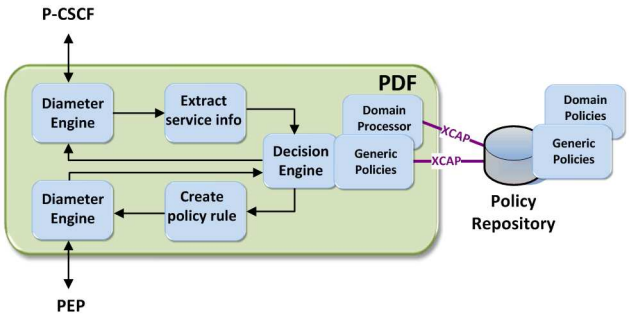

Figure 9: UCT PDF logical architecture.

layer. Each session has associated IP Flows that are defined by a 5 tuple and a QoS class; these are the dynamic rules that change as the IMS sessions themselves change and are enforced in the transport layer. The PDF monitors QoS provisioning through the receipt of transport layer events. These events result in session requests being sent to the control layer that can result in session termination or modification depending on the kind of event that was triggered. Fig. 9 shows the logical architecture of the UCT PDF.

Incorporated into the UCT PDF is a Web Management interface that is based on Java Server Pages. This connects to the XDM Policy Repository and allows for easy monitoring and editing of existing policies. Real time network monitoring is possible and dynamic IMS Session and IP Flow policy rules can be viewed as they are created or destroyed. The interface also manages the administration of the PDFs and PEPs, and allows an operator to define and monitor the network topology.

6.3 Policy Enforcement

The UCT PEP is a sub-element of the Policy and Charging Enforcement Point that enforces dynamic policy rules in the transport layer and monitors flow usage.

The element is Java based and implements the Diameter Gx interface to the PDF [2]. This interface is used to receive policy rules and to send transport layer events to the PDF. Upon receipt of the policy rules the UCT PEP translates the policy to transport layer specific configuration information. Linux Traffic Control is used to create a DiffServ router with several different service classes; this router acts as a gateway allowing only authorised IP Flows and depending on the definition of the policy rule the packets for each IP Flow are marked and queued accordingly. Fig. 10 shows the logical architecture of the UCT PEP.

Additionally the UCT PEP monitors the transport layer; a thread is created for each authorised IP Flow that monitors the bandwidth usage of that flow. Such feedback can be incorporated into future charging functionality and be used to calculate usage statistics or to detect transport layer events. The 3GPP defines several transport layer events [2], this architecture supports the Loss of Bearer and QoS Exceeds Authorisation events. A Loss of Bearer event could be triggered by an unexpected client crash, while a rogue client using more bandwidth than assigned could trigger a QoS

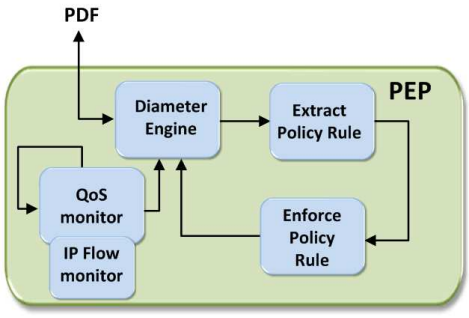

Figure 10: UCT PEP logical architecture.

Exceeds Authorisation event. Once triggered these events are sent to the PDF where they are processed and can result in session modification or termination.

Fig. 11 shows the signal flow for the detection of a Loss of Bearer event. When an unexpected client crash occurs the UCT PEP flow monitor detects this and triggers a Loss of Bearer event that is sent to the PDF. The PDF processes the event and sends a session request to the Application Function in the control layer, in this case the P-CSCF; the P-CSCF consequently deletes the session. Once receiving a positive response from the PDF, the UCT PEP reconfigures the transport layer device to block the relevant IP Flows. The P-CSCF sends an Abort Session Request to the PDF that results in policy rules being edited and resources being freed. Eventually a SIP BYE message is sent to the remaining client to indicate the Loss of Bearer to the end user.

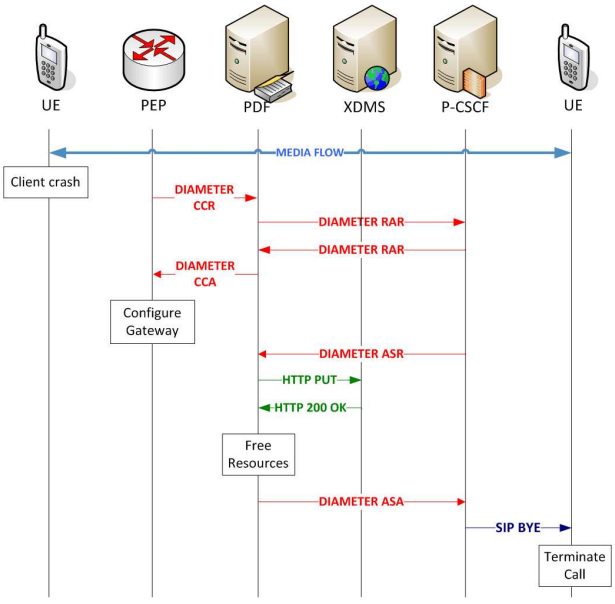

Figure 11: Loss of bearer signal flow.

6.4 Testbed Validation

The UCT Policy Control Framework works in co-ordination with the Open Source IMS Core; for validation tests the IMS Core and Policy Control Elements are run on a single Intel Pentium 4 PC with 768Mb RAM. The OpenXCAP XDM

Table 7: Average signalling overhead in the IMS core (kB).

Simultaneous requests	1	2	3	4	5
With QoS Provisioning	547.1	1175.9	1848.2	2627.3	3612.8
Without QoS Provisioning	38.2	71.6	109.9	144.3	180.8

implementation is run on a separate Intel P4 PC, while the UCT IMS Client runs on separate standard end point machines. By incorporating an RTP Proxy into the UCT PEP all media plane traffic is forced through this element for flow monitoring purposes. All testbed machines are connected via a 100 Mb/s LAN.

The increase in core signalling when setting up a session is examined to investigate the management overhead introduced by the Policy Control Framework. The increase in IMS core signalling when incorporating QoS provisioning depends on the session request and number of associated IP Flows; for a typical session with two IP Flows (one for audio and one for video) IMS Core signalling increases by 30 messages during session setup (These messages include additional Diameter requests and XCAP look-ups). Our hypothesis is that this traffic increase will be not be critical because the signalling is internal to the core and does not cause additional round trip delays - however with multiple simultaneous session requests it could become problematic.

Table 7 shows the signalling overhead at the IMS Core and Policy Control PC both with and without QoS provisioning enabled, ranging from a single session request to 5 simultaneous session requests. Signalling traffic was measured on all reference points between IMS Core elements and Policy Control elements. This gives an indication of how core signalling is affected when incorporating QoS provisioning. An overall increase in signalling overhead by roughly one order of magnitude is observed when utilising the Policy Control Architecture, it is interesting to note that the percentage increase in signalling overhead increases in proportion with the number of simultaneous session requests.

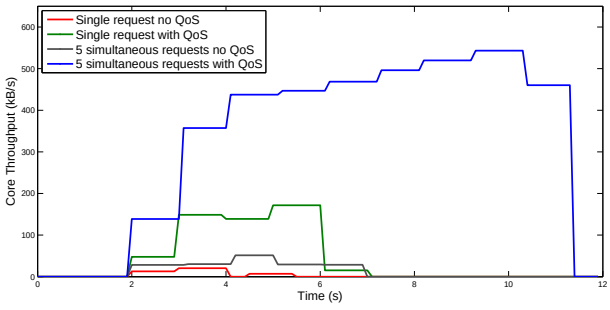

Figure 12: IMS Core throughput with and without QoS Provisioning.

Fig. 12 shows the throughput over 12 seconds at the IMS Core and Policy Control PC with and without QoS provisioning enabled for a single session request and for 5 simultaneous session requests. Signalling traffic was measured on all reference points between IMS Core elements and Policy Control elements. Again the order of magnitude core traffic increase is observed when provisioning QoS; furthermore it is clear that multiple session requests take significantly longer to be processed. In a commercial environment one could expect several thousands of session requests per second, such an increase in core signalling could cause an unacceptable increase in session setup delay. This problem could be further exasperated by increasing the geographical distance between the core elements involved. Further study is needed to determine the full effect of simultaneous QoS enabled session requests on session setup delay.

7. CONCLUSIONS

Four open-source IMS projects have been discussed and analysed. The UCT IMS Client provides a comprehensive emulation environment for the creation and deployment of IMS services. The UCT IPtv server is a platform for further investigation into IMS video streaming, while the UCT Conferencing Framework examines different teleconferencing solutions in the IMS context. The UCT Policy Control architecture defines a flexible environment for the easy creation and execution of policies that control network resources. These tools have been incorporated into a practical IMS test-bed based on the FOKUS Open Source IMS Core where they have been subject to real world testing and validation.

It is the intention of these open source initiatives to contribute to IMS development and encourage innovation in the field through the creation of open source and freely available practical development tools. As such the paper has shown both the capabilities of these tools as well as the limitations thus providing a relevant point of departure for future research in these fields.

8. REFERENCES

[1] 3GPP. TS 23.203 Policy and Charging Control Architecture. March 2007.

[2] 3GPP. TS 29.212 Policy and Charging Control over Gx reference point. 2007.

[3] 3GPP. TS 29.214 Policy and Charging Control over Rx reference point. 2007.

[4] 3GPP. TS 23.218 IP Multimedia (IM) session handling; IM call model; Stage 2. 2008.

[5] Mircea Amarascu. OpenXCAP. *http://www.openxcap.org/*.

[6] G. Camarillo and M.A. Garcia-Martin. *The 3G IP Multimedia Subsystem (IMS)*. John Wiley and Sons, 2nd edition, 2006.

[7] ETSI TiSPAN. ES 282 003 Resource and Admission Control Subsystem (RACS) Functional Architecture. 2006.

[8] Fraunhofer Institute FOKUS. Open Source IMS Core. *http://www.openimscore.org/*.

[9] PT Inovacao. IMS Communicator. *http://imscommunicator.berlios.de/*.

[10] Iptel.org. SIP Express Router. *http://www.iptel.org/ser/*.

[11] University of Cape Town. UCT IMS Client. *http://uctimsclient.berlios.de/*.

[12] Open Mobile Alliance. PoC XDM Specification version 1.0.2. September 2007.

[13] J. Rosenberg. RFC 4825 - The Extensible Markup Language Configuration Access Protocol. May 2007.

[14] Y. Snir, Y. Ramberg, J. Strassner, R. Cohen, and B. Moore. RFC 3644 - Policy Quality of Service Information Model. November 2003.

[15] D. Vingarzan and P. Weik. IMS Signaling over Current Wireless Networks: Experiments Using the Open IMS Core. *IEEE Vehicular Technology Magazine*, 2(1):28–34, March 2007.