

TransferHTTP+CAS: Latency and Memory Tests

¹Michael Adeyeye, ²Neco Ventura

¹Department of Information Technology, Cape Peninsula University of Technology, Cape Town
adeyem@cput.ac.za

²Communication Research Group, Department of Electrical Engineering
University of Cape Town
neco@crg.ee.uct.ac.za

Abstract-Convergence in the Internet, Telecommunication and Broadcasting is generating new services over the Internet. The online experience is getting improved via the introduction of user profile mobility and HTTP session mobility. While the academia has proposed different architectural schemes – client, proxy and server – for HTTP session mobility, the industry has introduced user profile mobility solutions, such as the Mozilla Weave and the Google Browser Sync. This paper presents our effort to improve the Web browsing experience via HTTP Session Mobility. A hybrid-based architecture is proposed and implemented here. It presents a distributed and centralized reference system for service creation. As a distributed reference system, it offers a multimedia service using Peer-to-Peer architecture, and as a centralized reference system, it offers controllable HTTP session mobility service using client-server architecture. The paper discusses the client (TransferHTTP) features, the proxy (CAS) features, our innovative multimedia service and the quality of experience of the system during HTTP session mobility with regards to memory consumption and latency.

I. INTRODUCTION

Quality of Experience (QoE) is a subjective measure of a customer's experience with its Communications Service Provider (CSP). CSPs could be classified into the telecommunications industry, which provides telephony-based services; the information industry, which provides the Internet-based services; and the entertainment industry, which provides the broadcast-based services [1].

QoE is subjective because experiences among customers would vary based on their cultural background, socioeconomic status, and personal experiences [2]. Early adopters of services will accept a product more based on the technology it uses and the function it performs than its ease of use and their experience in using it. The situation changes when a product ships to mainstream users. Normal users do not care nearly as much about what technology goes into a product. They care more about the problem the product solves and their experience while using it.

Convergence identifies a general pattern in the evolutionary process, namely the tendency to bring entities together, for example the coming together of classical telecommunications, the Internet, information technology and broadcasting, the ability to offer multiple services on a single network or the ability to offer the same service via more than one medium [3]. Interestingly, communications service providers (CSPs) may

soon no longer be the primary market drivers of communications services but increasingly drift towards a role of mediator and change-enabler [1].

With the convergence among the Internet, Broadcasting and Telecommunications, there are many technological possibilities and business models. This research uses SIP in its proposed hybrid-based architectural scheme to present the next stage of the Web, which will be – interactive applications, video, web sites collaborating and sharing information – all via the Web browser. The future of the Web as anticipated can only be achieved where the best of Internet protocols - SIP, HTTP and XML are used, and as shown in Fig. 1, our research project presents a reference system that offers Web session mobility between two Web browsers.

This paper reports the quality of experience of the HTTP session mobility service. The HTTP session mobility service is classified into content-sharing and session handoff service. For the regular Internet users, there are times they wish they could transfer existing web sessions between two PCs; two use case scenarios were mentioned in [4, 5]. The act of

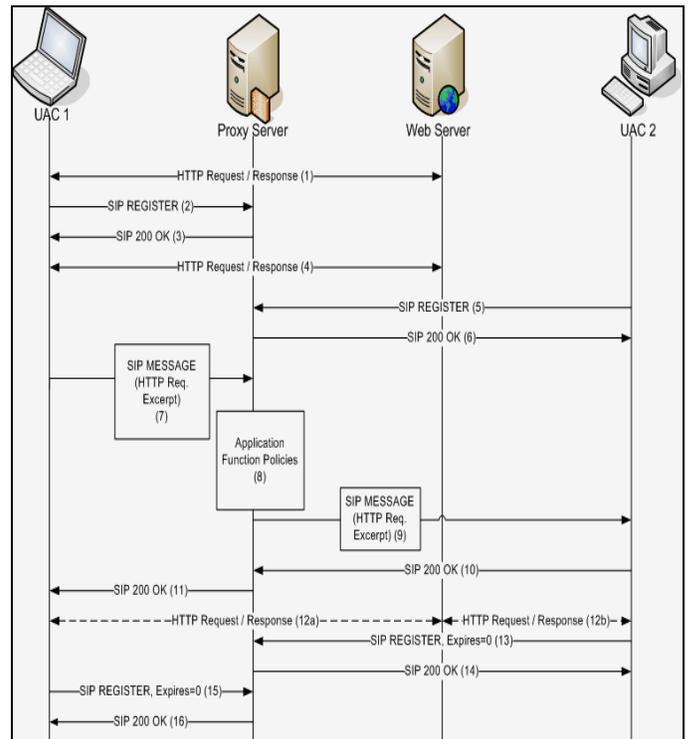


Figure 1. The Message Exchange between Web clients

referring another person to the same URL was called Content Sharing, while transferring existing web sessions between PCs was called Session Handoff. We have provided a fast and efficient way to accomplish both tasks within users' web browsers. The testbed in this work is made up of two Personal Computers (PCs), a Web server and a proxy. The metrics used include determining the latency introduced by the proxy server in Figure 1, when the system is used in client-server architecture. In addition, the latency between two PCs during a session transfer in a Peer-to-Peer architecture is also determined. The performance measurement for both architectures is determined when the PCs are connected to each other via the University Intranet and the Internet.

This paper is arranged as follows: Section II presents the Implementation Framework. Section III and IV present the client-side and proxy-side services, respectively. Section V discusses our experiences in the Implementation and significant contributions. Lastly, Section VI presents the conclusions and point at some future work.

II. THE IMPLEMENTATION

Figure 1 shows the message exchange between two PCs or Web clients. The architecture proposed and used here is a hybrid-based architecture [4, 5] that requires extending a Web browser and implementing a proxy server. The interaction could be between Web browsers running on PCs or smart devices and it transverses the proxy, which co-ordinates the interaction. The interaction, however, varies among Web browsers, based on users' policies and the kind of service chosen by the users. The following sub-sections present the client and the proxy features.

A. The Client Features

The Web browser extension is called TransferHTTP. The TransferHTTP extension added a new XPCOM with the contract id "@ngportal.com/SIPStack/SIPStackInit;1" into the existing XPCOMs in the browser. The extension core comprised of an XPCOM and an interface. The interface, named "ImyStack," was defined in an Interface Definition Language (IDL). Most of the built-in interfaces in Mozilla Firefox version 2.0 used in the implementation are nsIPref, nsICookie/nsICookieManager/nsICookieManager2, nsIThread, nsIPasswordManager/nsILoginManager, nsIIOService, nsIPromptService, nsITimer, nsIObserver.

The SIP stack for TransferHTTP ver. 1.4, which is a shared library, is 1.7MB in size, and it interacts with the extension via a Cross Platform Component Object Model (XPCOM). The extension adds a new menu called HTTP Mobility to the menubar in the Web browser. It has a "Preferences" submenu, which when clicked, gives users the opportunity to configure the Web browser. The settings include the SIP proxy address, SIP address and port number of the Web browser.

The SIP stack used in this implementation is the PJSIP [6], which is an Open Source project and small footprint

multimedia communication libraries written in C. The services available in the TransferHTTP extension include content sharing, session handoff, audio call and stream to call. They are chosen from the options in the statusbar, when the extension is installed. Figure 2 shows the new menu "HTTP Mobility" and available options in the statusbar, when TransferHTTP ver. 1.4 is installed.

Session handoff (also known as Transfer Session in Figure 2) is defined as the ability to move existing web session between two web browsers, and content sharing is defined as the ability to simultaneously view the same web page on two web browsers at the same time. Content sharing requires transferring only a web page Universal Resource Locator (URL).

An example is when Alice refers Bob to visit the same news website that she is browsing. In this case, she would only want the URL to be sent. Hence, she would have to choose the content sharing option. On the other hand, Session handoff requires the transfer of a web page's URL with its session data; for example, moving an email session between two Personal Computers (PCs). To continue checking an email without signing in again, the session handoff option will have to be chosen.

The "Make a call" option in Figure 2 enables a user to make a call to another, and the "Stream to Call" option enables a user to stream media to another user during a call session. The other options - Register Client, De-register Client and Accept Session - are used to register to a SIP Registrar, de-register the client and accept a Web session transfer request sent to the Web browser, respectively. The system uses Peer-to-Peer (P2P) architecture to stream media between users when the stream to call service is chosen, and client-server architecture is used during HTTP session mobility between users.

Various version of TransferHTTP extension are available. Version 1.2 was meant for Peer-to-Peer interaction, version 1.3 worked in a client-server environment, and current version

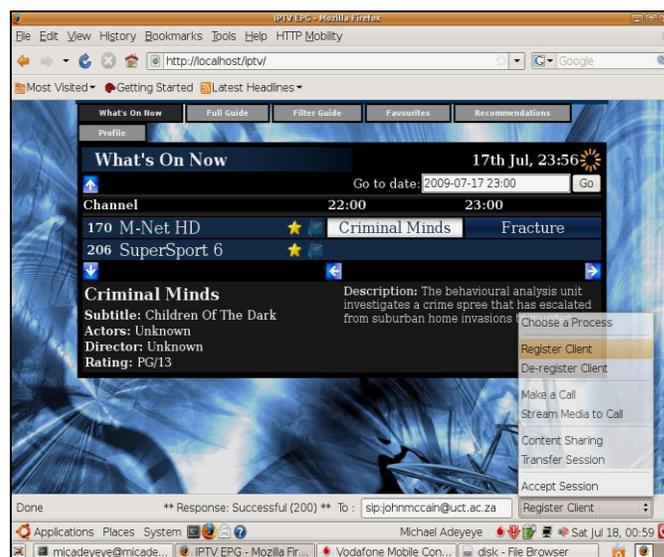


Figure 2. The Transfer Client User Interface

1.4 supports also VoIP and our novel the Stream Media to Call service. The browser extension, developed for the Mozilla Firefox open source browser, is publicly available for the SIP research community [7].

B. The Proxy Features

The Mobicents Communications Platform [8] was used to implement the proxy. It is an Open platform that enables creation, deployment and management of services and applications that integrate voice, video and data across a range of Internet Protocol (IP) and communications network by multiple devices. It implements and delivers both competing and interoperable programming models – JAIN SLEE and SIP SERVLETS – to develop Web and Voice over Internet Protocol (VOIP) applications that work together.

The services available on the proxy are Web session blocking and forwarding. Although applications could be developed using SIP Servlets or JSLEE in the Mobicents Platform, the current implementation is based on the Mobicents SIP Servlets Programming Model. When the proxy receives a SIP request, the Application Router (AR) in the server is called by a SIP container. The AR selects the appropriate SIP servlet application to service the SIP request. The SIP servlet application in our implementation responds to these requests – SIP INVITE, REGISTER and MESSAGE.

The Web browsers’ identities that need to be blocked or forwarded are currently hard-coded into the application. When a Web session transfer request is made, it is sent to the

application server. The application blocks or forwards the request based on its SIP address. How the Web session blocking and forward services work based on users’ policies is discussed in [9].

Figure 3 shows the proxy User Interface. The proxy logs all session transfer requests, call setup requests and actions taken on them. It provides the source SIP address, the destination SIP address, the SIP method, date, action taken (also known as status) and the referred URL in the case of a session transfer request. This information is available in the “Session Tracking and Pickup” page, as shown in Figure 3.

The “My Account” page enables a user to set his policies. Information available there includes his SIP URIs with their log-in details and policies. Here, he sets how every request should be handled. On the other hand, the “Buddy List” page contains a list of his contacts. He can also add a new contact via the page. He could also check the Buddy List page to see if his contacts are online or offline.

C. How the Stream Media to Call Service Works

The current implementation of the service is for two people in which media is streamed from one point to the other, regardless of who makes the call. Figure 2 shows an IPTV application. The Electronic Program Guide, in an Extensible Markup Language (XML) format, resides on the service provider’s PC. The service provider, here, refers to the person who hosts the audio/video files and the web contents.

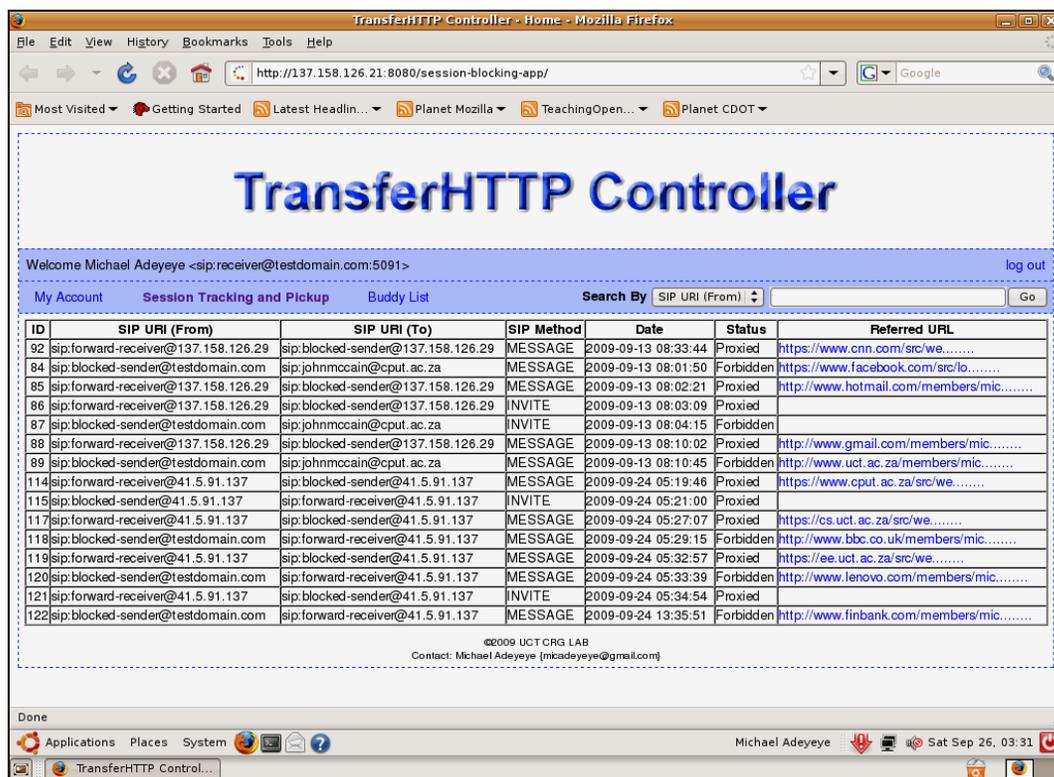


Figure 3. The TransferHTTP Proxy User Interface

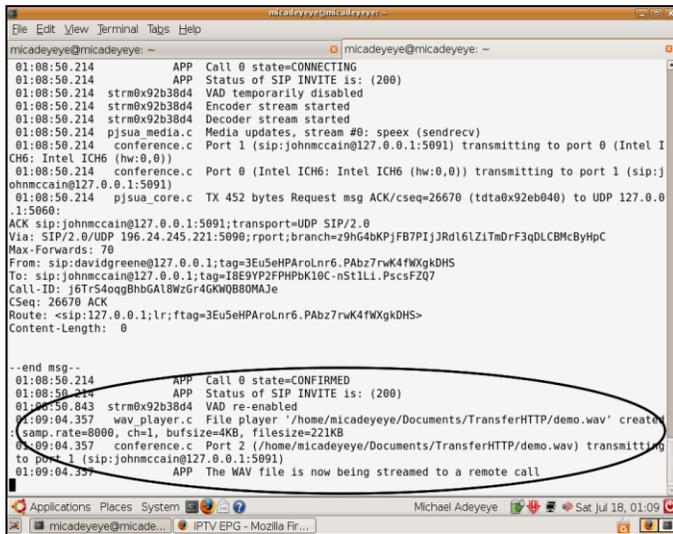


Figure 4. The Streaming of Media to a Call

The Web browser can act like a Web server when this extension [10] is installed. Hence, the visitor can access the Web contents on the service provider's PC. The service provider's PC can still act as a Web server when the extension is not installed. In this case, the provider is required to install necessary packages, such as a Web server and database engine.

To stream media, assume that the visitor has already established a call session by choosing the "Make a Call" option, the service provider is required to right-click on the image beside the yellow star and choose "Stream Media to Call" option. As shown in Figure 2, the yellow star is for the rating of the audio/video by the visitor(s). Figure 4 shows the SIP signalling, bridging of ports and streaming of media "demo.wav" from an absolute path to a callee, when the "Stream Media to Call" option is chosen.

The idea is that the service provider is providing a user-generated service in which anyone interested simply visits his website to see what programs are available. When a person is interested in a program, he is required to establish a call-session. The call session is automatically set-up with a 200 OK sent to the caller, and all that is required from the service provider is to right-click and choose the "stream to media" option. When the option is chosen, the physical path of the media is sent to the SIP stack, and the player in it is initiated.

The player, as the source, samples the file and creates a buffer. Thereafter, a conference bridge that connects the port of the source to the port of the destination (also known as the sink) is made. The audio/video can now be received at the destination. In this implementation, the service provider is only required to choose the stream to media option, as indicated in Figure 2, provided a call session already exists. Although this implementation streams media between two Web browsers, it could be extended to three or more Web browsers. The service could be extended to stream media to three or more web browsers thereby making it a broadcast service.

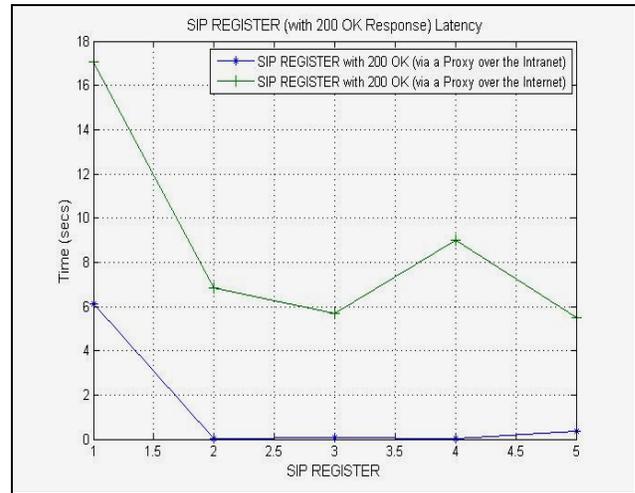


Figure 5. The SIP REGISTER (with 200 OK Rep.) Latency

III. THE TESTBED EXPERIMENT

The testbed is made of a laptop, referred to as UAC 1 in Figure 1, and two desktop PCs. One of the desktop PCs runs the Mobicents SIP AS, and the other, referred to as UAC 2 in Figure 1, has the TransferHTTP extension installed on it. The laptop also has the extension installed on it. The tests were conducted when all the PCs and laptop are connected over the University network (Intranet) and the Internet. In addition, the SIP AS was isolated during some of the tests. The isolation of the SIP AS made us carry out the tests in a Peer-to-Peer (P2P) environment, while having the SIP AS in the testbed during some of the tests made us carry out the tests in a Client-Server (C-S) environment.

The laptop and the desktop PC running the TransferHTTP extension were connected to the Internet via a 3G/HSDPA USB modem, when the test needed to be conducted via the Internet. The SIP AS has the application deployed and configured on it, as previously explained. The clocks on the laptop and the desktop PC that runs the extension were synchronized before the tests were conducted.

The tests included determining the latency in the signalling between UAC 1 and UAC 2. The message exchange is shown in Figure 4. The readings taken included time taken to register with the SIP AS, time taken by the SIP AS to block a session transfer request and time taken to send a request to another PC. When the tests were conducted in a P2P environment, the readings did not include time taken to register with the SIP AS or time taken to block a session transfer request. The reason is that the SIP AS was not required, so there was no registration of the clients with a server or blocking of requests sent from any of the clients.

However, the Web browsers, with the extension installed on them, were still able to transfer Web sessions, accepted Web sessions and establish multimedia sessions.

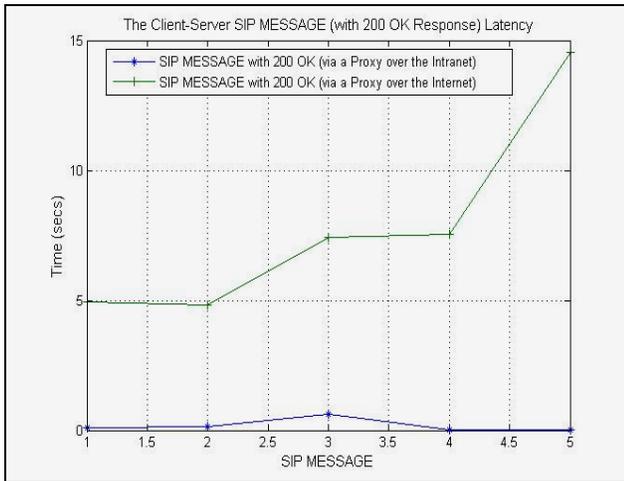


Figure 6. The C-S SIP MESSAGE (with 200 OK Rep.) Latency

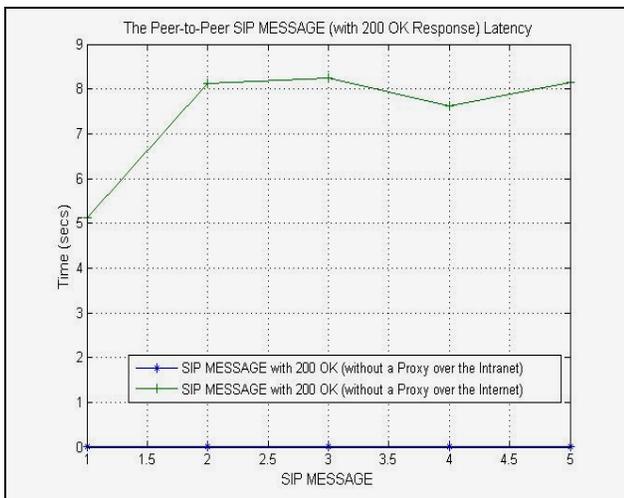


Figure 7. The P2P SIP MESSAGE (with 200 OK Rep.) Latency

IV. DISCUSSIONS

The use of Intranet and Internet in this section refers to conducting the test in a P2P environment and a client-server environment, respectively. Fig. 5 shows a graph of the readings taken when the clients register to the SIP AS over the Internet and the Intranet. Fig. 6 shows a graph of the readings taken when the clients successfully transfer sessions between each other via the SIP AS over the Internet and the Intranet. On the other hand, Fig. 7 shows a graph of the readings taken when the clients successfully transfer sessions between each other without the SIP AS over the Internet and the Intranet.

Fig. 8, however, shows a graph of the readings taken when session transfer requests were blocked by the SIP AS over the Internet and the Intranet. Note that there are no graphs of successful registration or blocking of session transfer requests

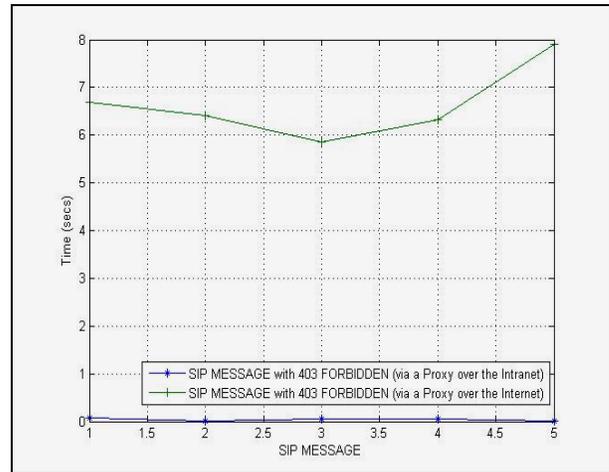


Figure 8. The C-S SIP MESSAGE (with 403 Forbidden Rep.) Latency

via the Internet and the Intranet when the SIP AS is not in use. The reason, as earlier explained, is that there was no SIP AS for the registration and co-ordination (i.e. blocking or forwarding) of session transfer requests when the tests were carried out in a P2P environment.

Regarding the connection speed, the 3G/HSDPA modem supports an average of 10.191Kbps for downlink and 1.962Kbps for uplink, while the University network supports an average of 125Kbps for downlink and 439Kbps for uplink. The readings were taken at different intervals for over two weeks; this helps us arrive at average values for the latency for each SIP method signalling.

The number of hops required to connect to the SIP AS over the Internet accounts for the high latency when the tests were carried out in a client-server environment. Figure 5 shows that the first registration with the SIP AS for both the Intranet and the Internet takes longest time than subsequent registrations. Figure 9 shows the memory consumption of the SIP AS when it was idle against when it was not idle. The reason is that the SIP AS, which is a JAVA application, goes into a “stand-by” mode when there is no SIP message to respond to in order to reduce the memory consumption on the PC. Hence, it takes some time for the SIP AS to respond to every first request. The SIP AS consumed 206MB of the 1GB RAM of the Pentium 4, 3GHz, 1GB RAM PC when it is in a stand-by mode. When the SIP AS was responding to a SIP message (SIP INVITE or SIP MESSAGE), its memory consumption averagely increased by 1.5MB.

Figures 5-8 show that the latencies, when the tests were carried out over the Intranet, were negligible. Henceforth, the discussions here will focus on the tests carried out over the Internet. Figure 7, when compared with Figure 6, shows that the latency in a P2P environment during a successful session transfer is higher than the latency in a C-S environment. The reason is that SIP AS already has the contact SIP Universal Resource Identifier (URI) of the destination, and it relays the session transfer request to the destination with little delay.

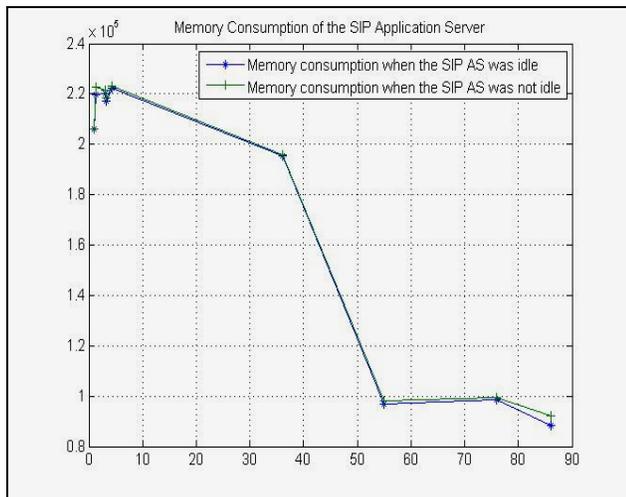


Figure 9. Memory Consumption of the SIP Application Server

When Figure 6 is compared with Figure 8, it would be noticed that the SIP AS introduced some more delay when blocking a request. Since Figures 6 and 8 are test results obtained in a C-S environment, it could be inferred that it takes longer time for the server to block or forbid request than to proxy or forward a request.

Table 1 shows the benchmark test for the proxy. It required using the pjsip-perf SIP performance measurement tool [11], which had a blocked SIP URI as its address. The pjsip-perf tool, which ran at the client end, sent 511 SIP INVITE messages at an average of 2055ms to the proxy, and the average response time of the proxy was 46373ms. The average number of SIP response messages was 541. The responses were SIP 403 Forbidden and SIP 408 Request Timeout (for messages that timed-out at proxy due to the delay caused by excessive overload). The reason the number of responses is higher than the number of requests is that SIP supports multiple responses to a request. The average memory consumption at the client was 7MB.

While the average memory consumption of the proxy per SIP message is 1.5MB (as shown in Figure 9), the benchmark test (as shown in Table 1) shows that the average memory consumption skyrockets to 13MB for 511 SIP messages in approximately 30secs. Considering the memory consumption during the benchmark-test, we could infer that CAS requires at most 300MB to run. The test results also show that the server could not block all requests. The proxy was only able to

Table 1. The Proxy performance under system overload

Test No.	Mem. Consumption (KB)	No. of 403 Forbidden Responses	No. of 408 Timeout Responses
1	14,236	96	407
2	9,336	101	406
3	11,948	83	434
4	14,568	96	404
5	17,090	88	410
Avg.	13,436	93	412

respond to an average number of 93 requests out of 511 requests sent to it in less than a minute. The client generated average of 412 Timeout responses when the proxy could not respond to the requests over a time of approximately 32secs. The proxy was however later seen generating the responses. It could be inferred that proxy bottleneck could be the number of Java threads devoted to servlet execution in Mobicents.

V. CONCLUSIONS

A new social network service for sharing URLs and Web sessions among peers has been discussed here. The performance results showed that the delay in an Intranet case is infinitesimal, while on the Internet, the system suffers some seconds delay. In addition, results also showed the services at the client and proxy are not memory intensive. This is a reference system for converged application development. The converged services – Web session blocking and forwarding, content sharing, session handoff and session tracking - require HTTP and SIP. Services in the proxy are called control services, and they are required to prevent abuse of the client services.

The obtained promising results are stimulating further research activities, and the client API, which exposes the signalling in Telecommunications to the Web, are available under a public license to create more innovative applications.

REFERENCES

- [1] Christian Saxtoft (2008), *Convergence: User Expectations, Communications Enablers and Business Opportunities*, John Wiley & Sons Ltd, England, pp. 34-37.
- [2] Ramesh Jain (2004), "Quality of Experience," *The IEEE Computer Society*, pp 96-97.
- [3] Hanrahan, H. (2007), *Network Convergence – Services, Applications, Transport, and Operations Support*, John Wiley & Sons, Ltd.
- [4] Michael Adeyeye, Neco Ventura and David Humphrey (2009), "Mapping Third Party Call Control and Session Handoff in SIP Mobility to Content Sharing and Session Handoff in the Web-browsing Context," in: *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC) 2009*, Budapest, April 5-8, 2009.
- [5] Michael Adeyeye (2008), *A SIP Integrated Web Browser for HTTP Session Mobility and Multimedia Services*, Unpublished Master's thesis, University of Cape Town, South Africa.
- [6] The PJSIP Stack, <http://www.pjsip.org>, July 31, 2009.
- [7] The TransferHTTP Web browser Extension, <http://transferhttp.mozdev.org>, September 5, 2008.
- [8] The Mobicents Open Source SLEE and SIP Server, <http://www.mobicents.org/index.html>, January 20, 2009.
- [9] Michael Adeyeye, Neco Ventura and David Humphrey (2009), "Control Services for the HTTP Session Mobility Service," in: *Proceedings of IEEE New Technologies, Mobility and Services (NTMS) 2009*, Egypt, December 20-23, 2009.
- [10] Mozilla Firefox Web Server Extension, <http://jamesboston.ca/cms/taxonomy/term/13>, July 31, 2009
- [11] PJSIP-PERF, http://www.pjsip.org/pjsip/docs/html/page_pjsip_perf_c.htm, October 26, 2009.

Michael Adeyeye is an instructor at the Cape Peninsula University of Technology. He is also a researcher at the University of Cape Town, where he is completing his Ph.D. His research interests include Multimodal and Multi-channel Access, Context-aware Services, Next Generation Networks Applications and Services and Open Standards efforts relating to mobility technologies and mobile application development.