Scalable Mobile Web Services Mediation Framework

Satish Srirama, Eero Vainikko, Vladimir Šor
Distributed Systems group
Institute of Computer Science, University of Tartu
J. Liivi 2, Tartu, Estonia
{srirama, eero, volli}@ut.ee

Matthias Jarke
Information Systems and Databases Group
RWTH Aachen University
Ahornstr. 55, 52056 Aachen, Germany
jarke@dbis.rwth-aachen.de

Abstract— **Web services are going mobile. A Mobile Enterprise can be established in a cellular network by participating Mobile Hosts, which act as web service providers, and their clients. Mobile Hosts enable seamless integration of user-specific services to the enterprise, by following web service standards, also on the radio link and via resource constrained smart phones. However, establishing such a Mobile Enterprise poses several technical challenges, like the quality of service (QoS) and discovery aspects, for the network and as well as for mobile phone users. The paper summarizes the challenges and research in this domain, along with our developed mobile web service mediation framework (MWSMF). We used a cloud computing infrastructure to setup one possible load balancing solution and also conducted number of tests to show that MWSMF is horizontally scalable. We also showed that elasticity of cloud platform provides a quick and easy manner to meet the load requirements of Mobile Enterprise.**

Keywords: Mobile web services, Mobile Host, Mobile Enterprise, cloud computing, QoS and enterprise service bus

I. INTRODUCTION

Mobile data services in tandem with web services [8] are seemingly the path breaking domain in current information systems research. In mobile web services domain, the resource constrained smart phones are used as both web service clients and providers (Mobile Host). Mobile terminals accessing the web services cater for anytime and anywhere access to services. Some interesting mobile web service applications are the provisioning of services like e-mail, information search, language translation, company news etc. for employees who travel regularly. There are also many public web services like the weather forecast, stock quotes etc. accessible from smart phones. Mobile web service clients are also significant in the geospatial and location based services [4]. While mobile web service clients are common, the scope of mobile web service provisioning (MWSP) was studied at RWTH Aachen University since 2003 [17], where Mobile Hosts were developed, capable of providing basic web services from smart phones. Mobile web service clients and the Mobile Hosts in a cellular network, together form a Mobile Enterprise.

Mobile Hosts enable seamless integration of user-specific services to the enterprise, by following standard web service interfaces and standards also on the radio link. Moreover, services provided by the Mobile Host can be integrated with larger enterprise services bringing added value to these services. For example, services can be provided to the mobile user based on his up-to-date user context. Context details like device and network capabilities, location details etc. can be obtained from the mobile at runtime and can be used in providing most relevant services like maps specific to devices and location information. Besides, Mobile Hosts can collaborate among themselves in scenarios like Collaborative Journalism and Mobile Host Co-learn System and bring value to the enterprise. [16]

Once the Mobile Host was developed, an extensive performance analysis was conducted to prove its technical feasibility [17]. While service delivery and management from Mobile Host were thus shown technically feasible, the ability to provide proper quality of service (QoS), especially in terms of security and reasonable scalability, for the Mobile Host is observed to be very critical. Similarly, huge number of web services possible, with each Mobile Host providing some services in the wireless network, makes the discovery of these services quite complex. Proper QoS and discovery mechanisms are required for successful adoption of mobile web services into commercial environments. Moreover, the QoS and discovery analysis of mobile web services has raised the necessity for intermediary nodes helping in the integration of Mobile Hosts with the enterprise. Based on these requirements a Mobile Web Services Mediation Framework (MWSMF) [19] is designed as an intermediary between the web service clients and the Mobile Hosts within the Mobile Enterprise, using the Enterprise Service Bus (ESB) technology.

While we were successful in establishing MWSMF on standard servers, the scale of the Mobile Enterprise is leading us to the new utility computing paradigm, cloud computing. In this study we are trying to establish the mediation framework on a public cloud infrastructure so that the framework can adapt itself to the loads of the mobile operator proprietary network, thus mainly helping in horizontal scaling and load balancing the MWSMF. The remaining sections of the paper are ordered as follows.

Section II discusses the details and challenges associated with providing services from smart phones and establishing a Mobile Enterprise. Section III discusses the details of the MWSMF. Section IV discusses cloud computing and load handling issues of the MWSMF along with the analysis and results. Section V concludes the paper with future research options.

II. MOBILE ENTERPRISE

A *Mobile Enterprise* can be established in a cellular network by participating Mobile Hosts and their clients, where the hosts provide user-specific services to the clients as per the WS* standards. However, such a Mobile Enterprise established, poses many technical challenges, both to the service providers and to the mobile operator. Some of the critical challenges and associated research are addressed in this section.

A. Challenges for establishing Mobile Enterprise

As the Mobile Host provides services to the Internet, devices should be safe from malicious attacks. For this, the Mobile Host has to provide only secure and reliable communication in the vulnerable and volatile mobile ad-hoc topologies. In terms of scalability, the Mobile Host has to process reasonable number of clients, over long durations, without failure and without seriously impeding normal functioning of the smart phone for the user.

Similarly, huge number of available web services, with each Mobile Host providing some services in the wireless network, makes the discovery of the most relevant services quite complex. Proper discovery mechanisms are required for successful adoption of Mobile Enterprise. The discovery, moreover, poses some critical questions like: where to publish the services provided by the Mobile Hosts? Should they be published with the centralized Universal Description, Discovery, and Integration (UDDI) registries available in the Internet or the operator is going to offer some help? This also raises questions like whether centralized nodes can withstand such high loads or some alternatives are to be looked at?

From the mobile operator's perspective the Mobile Enterprise poses questions like: what are the services expected by the mobile users from the operator? Can the operator monitor the communication and have a bird view of the complete network, so that business scenarios can be drawn out of it? Do operators have such infrastructure that can scale and adapt to such huge oscillating requirements? What about the scalability of such infrastructure?

Our research in this domain focused at addressing most of these issues and the remaining parts of this paper summarize the research and results.

B. QoS aspects of the Mobile Host

Providing proper QoS, especially, appropriate security and reasonable scalability, for mobile web service provisioning domain was observed to be very critical. The security analysis of the Mobile Host studied the adaptability of WS-Security specification to the MWSP domain and concludes that not all of the specification can be applied to the Mobile Host, mainly because of resource limitations. The results of our analysis suggest that the mobile web service messages of reasonable size, approximately 2-5kb, can be secured with web service security standard specifications. The security delays caused are approximately 3-5 seconds. We could also conclude from the analysis that the best way of securing messages in a Mobile Enterprise is to use AES (Advanced Encryption Standard) symmetric encryption with 256 bit key, and to exchange the keys with RSA 1024 bit asymmetric key exchange mechanism and signing the messages with RSAwithSHA1. But there are still high performance penalties when messages are both encrypted and signed. So we suggest encrypting only the parts of the message, which are critical in terms of security and signing the message. The signing on top of the encryption can completely be avoided in specific applications with lower security requirements. [18]

In terms of scalability, the layered model of web service communication, introduces a lot of message overhead to the exchanged verbose XML based SOAP messages. This consumes a lot of resources, since all this additional information has to be exchanged over the radio link. Thus for improving scalability the messages are to be compressed without effecting the interoperability of the mobile web services. Message compression also improves the energy efficiency of the devices as there will be less data to transmit.

In the scalability analysis of the Most Host, we have adapted BinXML [6] for compressing the mobile web service messages. BinXML is a light-weight XML compression mechanism, which replaces each XML tag and attribute with a unique byte value and replaces each end tag with 0xFF. By using a state machine and 6 special byte values including 0xFF, any XML data with circa 245 tags can be represented in this format. The approach is specifically designed to target SOAP messages across radio links. So the mobile web service messages are exchanged in the BinXML format, and this has reduced the message of some of the services by ~30%, drastically reducing the transmission delays of mobile web service invocation. The BinXML compression ratio is very significant where the SOAP message has repeated tags and deep structure. The binary encoding is also significant for the security analysis as there was a linear increase in the size of the message with the security incorporation. The variation in the WS-Security encrypted message size for a typical 5 Kb message is approximately 50%. [16]

C. Discovery aspects of the Mobile Enterprise

In a commercial Mobile Enterprise with Mobile Hosts, and with each Mobile Host providing some services for the Internet, expected number of services to be published could be quite high. Generally web services are published by advertising WSDL (Web Services Description Language) descriptions in a UDDI registry. But with huge number of services possible with Mobile Hosts, a centralized solution is not the best idea, as they can have bottlenecks and can introduce single points of failure. Besides, mobile networks are quite dynamic due to the node movement. Devices can join or leave network at any time and can switch from one operator to another operator. This makes the binding information in the WSDL documents, inappropriate. Hence the services are to be republished every time the Mobile Host changes the network.

Dynamic service discovery is one of the most extensively explored research topics in the recent times. Most of these service discovery protocols are based on the announce-listen model like in Jini. In this model periodic multicast mechanism is used for service announcement and discovery. But these mechanisms assume a service proxy object that

acts as the registry and it is always available. For dynamic ad hoc networks, assuming the existence of devices that are stable and powerful enough to play the role of the central service registries is inappropriate. Hence services distributed in the ad-hoc networks must be discovered without a centralized registry and should be able to support spontaneous peer to peer (P2P) connectivity. [5] proposes a distributed peer to peer Web service registry solution based on lightweight Web service profiles. They have developed VISR (View based Integration of Web Service Registries) as a peer to peer architecture for distributed Web service registry. Similarly Konark service discovery protocol [12] was designed for discovery and delivery of device independent services in ad hoc networks.

Considering these developments and our need for distributed registry and dynamic discovery, we have studied alternative means of mobile web service discovery and realized a discovery mechanism in the P2P network. In this solution, the virtual P2P network also called the mobile P2P network is established in the mobile operator network with one of the nodes in operator proprietary network, acting as a JXTA super peer. JXTA (Juxtapose) is an open source P2P protocol specification. Once the virtual P2P network is established, the services deployed on Mobile Host in the JXME virtual P2P network are to be published as JXTA advertisements, so that they can be sensed as JXTA services among other peers. JXTA specifies *Modules* as a generic abstraction that allows peers to describe and instantiate any type of implementation of behavior representing any piece of "code" in the JXTA world. So the mobile web services are published as JXTA modules in the virtual P2P network. Once published to the mobile P2P network, the services can later be discovered by using the keyword based search provided by JXTA. This approach also considered categorizing the services and the advanced features like context aware service discovery. We address the discovery solution as mobile P2P discovery mechanism. The evaluation of the discovery approach suggested that the smart phones are successful in identifying the services in the P2P network, with reasonable performance penalties for the Mobile Host. [20]

III. MOBILE WEB SERVICES MEDIATION FRAMEWORK

Mobile Hosts with proper QoS and discovery mechanisms, enable seamless integration of user-specific services to the Mobile Enterprise. Moreover services provided by the Mobile Host can be integrated with larger enterprise services bringing added value to these services. However, enterprise networks deploy disparate applications, platforms, and business processes that need to communicate or exchange data with each other or in this specific scenario addressed by the paper, with the Mobile Hosts. The applications, platforms and processes of enterprise networks generally have non-compatible data formats and non-compatible communications protocols. Besides, within the domain of our research, the QoS and discovery study of the Mobile Host offered solutions in disparate technologies like JXTA. This leads to serious integration problems within the networks. The integration problem extends further if two or more of such enterprise networks have to communicate among themselves. We generally address this research scope and domain, as the *Enterprise Service Integration*.

The mobile web services mediation framework (MWSMF) is established as an intermediary between the web service clients and the Mobile Hosts in mobile enterprise. ESB is used as the background technology in realizing the mediation framework. Similar mediation mechanisms for mobile web services are addressed in [11]. Especially, [11] describes the status of research with provisioning services from resource constrained devices. When considering mediation within semantic web services, Web Service Modeling Ontology (WSMO) has significant contributions [13]. However, we went with the ESB approach, due to the availability of several open source implementations.

Figure 1 shows the Mobile Enterprise and the basic components of the mediation framework. For realizing the mediation framework we relied on ServiceMix [3], an open source implementation of ESB, based on the JBI specification [22]. JBI architecture supports two types of components Service Engines and Binding Components. Service engines are components responsible for implementing business logic and they can be service providers/consumers. Service engine components support content-based routing, orchestration, rules, data transformations etc. Service engines communicate with the system by exchanging normalized messages across the normalized message router (NMR). The normalized messaging model is based on WSDL specification. The service engine components are shown as straight lined rectangles in the figure. Binding components are used to send and receive messages across specific protocols and transports. The binding components marshall and unmarshall messages to and from protocol-specific data formats to normalized messages. The binding components are shown as dashed rectangles in the Figure 1.

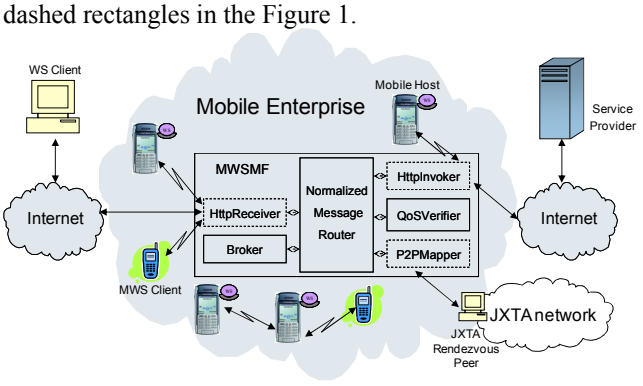

Figure 1. Mobile Enterprise setup with Mobile Hosts and MWS clients

The HttpReceiver component shown in figure 1 receives the web service requests (SOAP over HTTP) over a specific port and forwards them to the Broker component via NMR. The main integration logic of the mediation framework is maintained at the Broker component. For example, in case of the scalability maintenance, the messages received by Broker are verified for mobile web service messages. If the

messages are normal Http requests, they are handled by the HttpInvoker binding component. If they comprise mobile web service messages, the Broker component further ensures the QoS of the mobile web service messages and transforms them as and when necessary, using the QoSVerifier service engine component, and routes the messages, based on their content, to the respective Mobile Hosts. The framework also ensures that once the mobile P2P network is established, the web service clients can discover the services using mobile P2P discovery mechanism and can access deployed services across MWSMF and JXTA network. [19]

Apart from security and improvements to the scalability, QoS provisioning features of the MWSMF also include message persistence, guaranteed delivery, failure handling and transaction support. External web service clients, that do not participate in the mobile P2P network, can also directly access the services deployed on the Mobile Hosts via MWSMF, as long as the web services are published with any public UDDI registry or the registry deployed at the mediation framework and the Mobile Hosts are provided with public IPs. This approach evades the JXME network completely. Thus the mediation framework acts as an external gateway from Internet to the Mobile Hosts and mobile P2P network. The framework also provides a bird view of the mobile enterprise to the cellular operator, so that business scenarios can be drawn out of it. Preliminary analysis of the mediation framework is available at [16].

IV. MWSMF ON THE CLOUD

While the MWSMF was successful in achieving the integrational requirements of the Mobile Host and the Mobile Enterprise, a standalone framework again faces the troubles with heavy loads. The problems with scalability are quite relevant in such scenarios and the system should scale on demand. For example number of Mobile Hosts providing the services and the number of services provided by the Mobile Hosts can explode while some events are underway; like Olympics or national elections etc. Some of these application scenarios are addressed in [16]. This increases the number of MWS clients the framework has to support. Elasticity of the framework can be defined as its ability to adjust according to the varying number of requests, it has to support. As the study targets the scales of mobile operator proprietary networks, to achieve elasticity, horizontal scaling and load balancing for the MWSMF, we tried to realize the mediation framework on a public cloud.

A. *Cloud computing*

Cloud computing is a style of computing in which, typically, real-time scalable resources are provided "as a service (aaS)" over the Internet to users who need not have knowledge of, expertise in, or control over the cloud infrastructure that supports them. The provisioning of cloud services can be at the Infrastructural level (IaaS) or Platform level (PaaS) or at the Software level (SaaS). A cloud computing platform dynamically provisions, configures, reconfigures, and de-provisions servers as requested. Cloud computing mainly forwards utility computing model, where consumers pay based on their usage. Cloud computing also benefits from economies of scale. Servers in the cloud can be physical machines or virtual machines.

While there are several public clouds on the market, Google Apps (example of SaaS, includes Google Mail, Docs, Sites, Calendar, etc), Google App Engine (example of PaaS, provides elastic platform for Java and Python applications with some limitations) and Amazon EC2 (example of IaaS) are probably most known and widely used. Elastic Java Virtual Machine on Google App Engine allows developers to concentrate on creating functionality rather than bother about maintenance and system setup. Such sandboxing, however, places some restrictions on the allowed functionality [8]. Amazon EC2 [1] on the other hand allows full control over virtual machine, starting from the operating system. It is possible to select a suitable operating system, and platform (32 and 64 bit) from many available Amazon Machine Images (AMI) and several possible virtual machines, which differ in CPU power, memory and disk space. This functionality allows freely select suitable technologies for any particular task. In case of Amazon EC2 price for the service depends on machine size, its uptime, and used bandwidth in and out of the cloud.

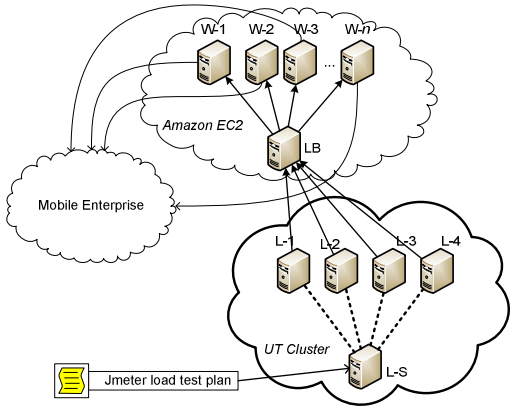

Figure 2. Load test setup for the MWSMF

Flexibility of EC2 environment and existing MWSMF implementation were some of the reasons EC2 was chosen for the experiment. Moreover, there are free implementations of EC2 compatible cloud infrastructure – Eucalyptus [7]. Eucalyptus allows creating private clouds compatible with Amazon EC2. Our research group is in the process of setting up a scientific computing cloud (SciCloud) on one of our clusters, using Eucalyptus technology. With this cloud, students and researchers can efficiently use the already existing resources of university computer networks, in solving computationally intensive scientific, mathematical, and academic problems [15].

B. *Load balancing the MWSMF from the cloud*

To achieve the scalability for the mediation framework, the MWSMF was installed on the Amazon EC2 cloud. Once the Amazon Machine Images (AMI) are configured, stateless nature of the MWSMF allows, fairly easy horizontal scaling by adding more MWSMF nodes and distributing the load among them with the load balancer. Figure 2 shows the deployment scenario with the load balancer (LB) and the MWSMF worker AMI nodes (W-1, W-2, W-3, and W-*n*).

There are several load balancing techniques that can be used in this scenario. One approach is to use DNS based load balancing, where each call to the DNS server will result in different IP address. This means that each MWSMF node will be accessed by certain subset of clients directly, without an intermediary load balancing proxy as discussed below. This approach is not fault tolerant in case the framework node would crash but its IP would be cached on the client's DNS cache. However, this approach is inevitable, if loads on the single proxy based load balancer will grow to a level that a single load balancer itself will become a bottleneck. Another approach is to use load balancing proxy server in front of MWSMF nodes. Among other options, Apache HTTPD server with mod_proxy and mod_load_balancer is probably most commonly used configuration. It has one major drawback in elastic environment, as it doesn't allow dynamic reconfiguration of worker nodes. If we add or remove some MWSMF nodes we are required to restart load balancer as well, which is not convenient and potentially introduces some failed requests during restart. Alternative http proxy load balancer HAProxy [21] allows such dynamic behavior. However we used Apache HTTP server with mod_proxy and mod_load_balancer [2] as a load balancer for the MWSMF because it is more widespread and we had experience in configuring such setup, which was important, as the aim of this research was to show the horizontal scalability rather than achieve maximum automation.

C. *Scalability of the MWSMF*

Load testing of MWSMF on the cloud was performed in a distributed manner using JMeter – open source load testing software. Figure 2 shows the deployment setup in detail. JMeter was deployed on one of the clusters in the University of Tartu (UT Cluster). Deployment consisted of 4 slave nodes (L-1, L-2, L-3, and L-4) and 1 master node (L-S). Load testing scenario (called a test plan in Jmeter) is loaded on the master node, which sends it to the slave nodes and initiates load testing. During the test run each slave node sends testing results back to the master node, where results are aggregated into a single report.

In our test scenario we performed 5 consequent requests by n concurrent threads, where n varied between 75 and 250 per slave node, which makes 300 to 1000 concurrent requests on a load balancer, thus simulating a large number of simultaneous clients for the MWSMF and the Mobile Host in Mobile Enterprise. Another important factor that impacts test results is a connection and response timeout on the client, in our test case – the slave node. Connection timeout is a time until connection to the server is established and response timeout is the time since call starts on the client side until response is received. If these timeouts are long enough, then observations showed, that even single MWSMF node can withstand large loads, due to the sufficient QoS of the ESB. However, in such scenario a call may last too long for a mobile client ant it may start retransmitting instead of waiting. In our tests we set connection-response timeout to 50-70 seconds. It must be also noted that, in the real life connection timeout on a client side is not a parameter that the service provider can affect nor predict. In case of interactive applications, where user interaction is involved, response should be preferably delivered in less than 10 seconds to keep user's attention [14].

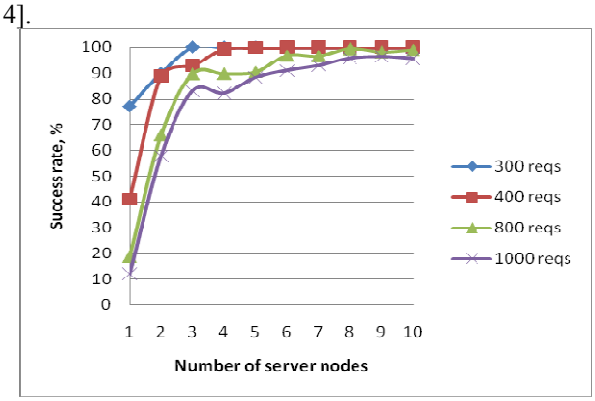

Figure 3. Success rate of concurrent requests over multiple server nodes

On the cloud front a load balancer (LB) and up to 10 MWSMF worker nodes were set up. To show the elasticity of the cloud we increased the number of the server nodes from 1 to 10 after each test. All servers were running on Amazon EC2 infrastructure and all of them were using EC2 small instances. Small instance has 1.7 GB of memory, CPU power of 1 EC2 Compute unit, which is equivalent to CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor as of 07.12.2009 (CPU capacity of an EC2 compute unit do change in time). Both load balancer and worker nodes were running 32 bit Linux platforms. Apache HTTP server version 2.2.8 with mod_proxy and mod_load_balancer were used as a load balancer. Load balancer was setup to use request based scheduling, which means that all worker nodes received equal amount of requests. However, it is possible to configure load balancer based on traffic or busyness. Busyness means how many concurrent requests a worker node has at the moment the new request arrives. In real-life situation best load balancing algorithm for a particular scenario should be chosen based on the services provided by the mobile enterprise and the nature of the request/response traffic. For more details on load balancing algorithm refer [2].

In the load test of the MWSMF, we measured how success rate of the requests depends on a number of worker nodes depending on a number of concurrent requests. Success means that a request will get a response before connection or response timeout occurs and success rate shows how many requests from all performed requests succeeded. The results of the experiment are shown in figure 3. From the diagram it can be clearly seen that the percentage of succeeded requests grows logarithmically with the number of nodes and degrades exponentially as load grows. Performance of a single node drops rapidly already after 300 concurrent requests and even with 300 concurrent requests success rate is only 77%, however 3 nodes can handle this load with 100% success rate. It can be also seen, that with current setup adding more nodes does not show any visible effect after 6 nodes and performance is improved by an insignificant fraction in contrast to difference between 1, 2 and 3 nodes. In summary we observed that, with current MWSMF implementation one single node can handle around

100-130 concurrent MWS requests with 100% success rate. Adding an additional node adds roughly 100 new concurrent requests to the total capability until the load grows up to 800 concurrent requests, when load balancer itself becomes a bottleneck and adding any additional nodes do not give desired effect. This analysis showed mediation framework to be horizontally scalable. However, certain loads demand more advanced load balancing techniques. The elastic cloud environment helps to achieve this required setup very quickly.

V. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

The developments in the web services domain, the improved device capabilities of the smart phones and the improved transmission capabilities of the cellular networks have lead to the mobile web services provisioning domain. With this paper, we tried to summarize the challenges and research associated in this domain and establishing the Mobile Enterprise. The QoS aspects of the developed Mobile Host, like providing proper security and scalability, and the discovery of the provided services are addressed briefly. Further, the QoS and discovery analyses of the Mobile Host have raised the necessity for a middleware framework and the features and realization details of the MWSMF are discussed. This paper showed that MWSMF is horizontally scalable, thus allowing to utilize cloud's elasticity to meet load requirements in an easy and quick manner.

In these experiments we configured load balancer manually and one of future research directions is to achieve more automation in scaling process – detect loads automatically, dynamically add more working nodes and automatically configure load balancer to accommodate new worker nodes. After loads drop, dynamically scalable MWSMF should shutdown unnecessary worker nodes. Another future research direction is to use Eucalyptus framework for cloud infrastructure instead of Amazon EC2, to show that public cloud's elasticity is achievable also in private clouds. We also want to extend this experience to our scientific computing cloud (SciCloud) project.

ACKNOWLEDGMENT

The research is supported by the Estonian Science Foundation under Mobilitas program and the European Regional Development Fund through the Estonian Centre of Excellence in Computer Science. The work was earlier supported by the Ultra High-Speed Mobile Information and Communication (UMIC) research cluster at RWTH Aachen University. The authors would also like to thank R. Levenshteyn and M. Gerdes of Ericsson Research for their help and support.

REFERENCES

[1] Amazon Inc, "Amazon Elastic Compute Cloud (Amazon EC2)", http://aws.amazon.com/ec2/ 10.12.2009.

[2] Apache Software Foundation, "Apache Module mod_proxy_balancr", http://httpd.apache.org/docs/2.2/mod/mod_proxy_balancer.html 10.12.2009.

[3] Apache Software Foundation, "Apache ServiceMix", http://incubator.apache.org/servicemix/home.html 10.12.2009.

[4] B. Benatallah and Z. Maamar, "Introduction to the special issue on m-services", IEEE transactions on systems, man, and cybernetics - part a: systems and humans, 33(6):665–666, November 2003.

[5] S. Dustdar and M. Treiber, "Integration of transient Web services into a virtual peer to peer Web service registry", Distributed and Parallel Databases, 20: 91–115, 2006.

[6] M. Ericsson and R. Levenshteyn, "On optimization of XML-based messaging", In Second Nordic Conference on Web Services (NCWS 2003), pages 167–179, 2003.

[7] Eucalyptus Systems Inc, "Eucalyptus", http://www.eucalyptus.com/ 10.12.2009.

[8] Google Inc, "App Engine Java Overview", http://code.google.com/appengine/docs/java/overview.html 10.12.2009.

[9] K. Gottschalk, S. Graham, H. Kreger, and J. Snell, "Introduction to web services architecture", IBM Systems Journal: New Developments in Web Services and E-commerce, 41(2):178–198, 2002.

[10] M. Keen, S. Bishop, A. Hopkins, S. Milinski, C. Nott, and R. Robinson, "Patterns: Implementing an SOA using an Enterprise Service Bus", IBM RedBooks, July 2004.

[11] Y. Kim and K. Lee, "A lightweight framework for mobile web services", Journal on Computer Science - Research and Development, 24(4):199-209, Springer.

[12] C. Lee, A. Helal, N. Desai, V. Verma, and B. Arslan, "Konark: A System and Protocols for Device Independent, Peer-to-Peer Discovery and Delivery of Mobile Services", IEEE transactions on systems, man, and cybernetics - part a: systems and humans, 33(6):682–696, 2003.

[13] A. Mocan, E. Cimpian, M. Stollberg, F. Scharffe, and J. Scicluna, "WSMO Mediators", December 2005. http://www.wsmo.org/TR/d29/ 10.12.2009

[14] J. Nielsen, "Usability Engineering", Morgan Kaufmann, San Francisco, 1994.

[15] S. N. Srirama, "Scientific Computing on the Cloud (SciCloud)", http://ds.cs.ut.ee/research/scicloud 10.12.2009.

[16] S. N. Srirama and M. Jarke, "Mobile Hosts in enterprise service integration", International Journal of Web Engineering and Technology (IJWET), 5(2):187-213, 2009. Inderscience Publishers.

[17] S. N. Srirama, M. Jarke, and W. Prinz, "Mobile web service provisioning", In AICT-ICIW '06: Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services, page 120. IEEE Computer Society, 2006.

[18] S. N. Srirama, M. Jarke, and W. Prinz, "Security analysis of mobile web service provisioning", International Journal of Internet Technology and Secured Transactions (IJITST), 1(1/2):151–171, Inderscience publishers, 2007.

[19] S. N. Srirama, M. Jarke, and W. Prinz, "MWSMF: A Mediation Framework Realizing Scalable Mobile Web Service Provisioning", In International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications (Mobilware 2008), Innsbruck, Austria, ACM Int. Conf. Proceeding Series; Vol. 278, Article No. 43, 2008.

[20] S. N. Srirama, M. Jarke, and W. Prinz and H. Zhu, "Scalable Mobile Web Service Discovery in Peer to Peer Networks", In IEEE Third International Conference on Internet and Web Applications and Services (ICIW 2008), Pages 668-674, IEEE Computer Society, 2008.

[21] W. Tarreau, "HAProxy Architecture Guide, version 1.1.34", Jan, 2006, http://haproxy.1wt.eu/download/1.3/doc/architecture.txt 10.12.2009.

[22] R. Ten-Hove and P. Walker, "Java TM Business Integration (JBI) 1.0 -JSR 208", Final Release, Technical report, Sun Microsystems Inc., 2005.