Scaling Mobile Enterprise Through Cloud Computing

Satish Narayana Srirama, Eero Vainikko
Distributed Systems group
Institute of Computer Science, University of Tartu
J. Liivi 2, Tartu, Estonia
{srirama, eero}@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 (WS) are going mobile. A Mobile Enterprise can be established in a cellular network by participating Mobile Hosts, which act as WS providers, and their clients. Mobile Hosts enable seamless integration of user-specific services to the enterprise, by following WS standards, also on the radio. The paper summarizes the challenges and research associated with establishing Mobile Enterprise, along with our developed mobile web service mediation framework (MWSMF). However, to scale Mobile Enterprise to the loads possible in cellular networks, we tried to shift some of its components to the new utility computing paradigm, cloud computing. The cloud based load balancing for the Mobile Enterprise can be provided at the middleware framework level or at the individual services level. This paper described both the approaches with application scenarios. The analysis concludes that MWSMF and its components are horizontally scalable, thus allowing to utilize elasticity of cloud platform to meet load requirements of Mobile Enterprise.**

Keywords: Mobile web services, Mobile Host, Mobile Enterprise, cloud computing, QoS and load balancing

I. INTRODUCTION

Cloud computing [4] is a style of computing in which, typically, resources scalable on demand, 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. This ensures elasticity of the systems deployed in the cloud. Cloud computing mainly forwards utility computing model, where consumers pay based on their usage. Servers in the cloud can be physical or virtual machines. [4]

Besides, web services (WS) [8] are going mobile. In mobile web services (MWS) domain, the resource constrained smart phones are used as both WS clients and providers (Mobile Host). Some interesting MWS client applications are the provisioning of services like e-mail, search, language translation, company news etc. for employees who travel regularly. There are also many public WS like the weather forecast, stock quotes etc. accessible from smart phones [5]. While MWS clients are common, the scope of mobile web service provisioning (MWSP) was studied at RWTH Aachen University since 2003 [13], where Mobile Hosts were developed, capable of providing basic WS from smart phones. Mobile Hosts thus enable pervasive computing and services. Several application of Mobile Host, that bring value to the enterprise were demonstrated [11]. MWS clients and the Mobile Hosts in a cellular network, together form a *Mobile Enterprise*. Establishing such a Mobile Enterprise poses several technical challenges, both to the service providers and mobile operator.

While we were successful in establishing Mobile Enterprise, the scale and dynamic loads of Mobile Enterprise has lead us to the cloud computing paradigm. We also have observed load balancing to be the key in successful deployment of Mobile Enterprise in commercial environments. Hence to scale the system on demand and to provide the load balancing for the Mobile Enterprise, we tried to establish the middleware components of the system on a public cloud infrastructure.

There are several public clouds available on the market. Google App Engine [7] (provides elastic platform for Java and Python applications) and Amazon Elastic Compute Cloud (EC2) [1] are probably most known and widely used. EC2 allows full control over virtual machine, starting from the operating system (OS). It is possible to select a suitable OS, 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. These features of EC2 allow to freely selecting suitable technologies for any particular task. In case of EC2, price for the service depends on machine size, its uptime, and used bandwidth in and out of the cloud. Considering the flexibility of EC2 environment and our existing Mobile Enterprise implementation, we proceeded with EC2.

The remaining sections of the paper are ordered as follows. Section II discusses the challenges associated with providing services from smart phones and establishing a Mobile Enterprise. Section III discusses the details of scaling Mobile Enterprise with cloud computing. Section IV concludes the paper with future research options.

II. MOBILE ENTERPRISE AND RESEARCH CHALLENGES

A *Mobile Enterprise* [12], as shown in figure 1, 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. For example, the applications hosted on a Mobile Host provide information about the associated person (e.g. location,

agenda) as well as the surrounding environment (e.g. signal strength, bandwidth). Mobile devices also support multiple integrated devices (e.g. camera) and auxiliary devices (e.g. GPS receivers, printers). The hosted services can use them in providing valuable information to the outside world. [11]

Moreover, services provided by the Mobile Host can be integrated with larger enterprise 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. Several applications of the Mobile Host are designed and developed like the MobileHost CoLearn System, Collaborative Journalism and remote patient tele-monitoring. Mobile Host feature also allows deploying new services and software dynamically on the smart phones. This over the air deployment capability enabled us in providing pervasive services and building distributed computing frameworks from smart phones. [11]

While applications possible with Mobile Host are interesting, providing proper QoS, especially, appropriate security and reasonable scalability, is very critical. 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 reliable communication in the volatile mobile ad-hoc topologies. The security analysis of the Mobile Host studied the adaptability of WS-Security specification to the MWSP domain. The analysis suggests that the MWS messages of reasonable size, approximately 2-5kb, can be secured with WS security standard specifications. [12]

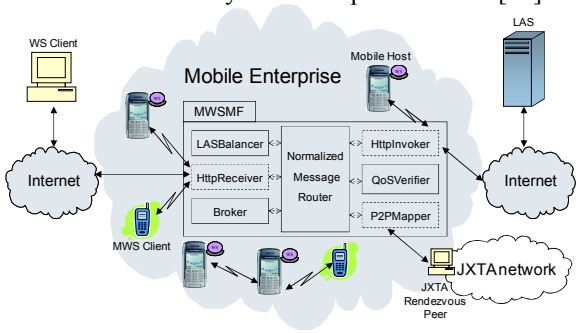

Figure 1. Mobile Enterprise setup with Mobile Hosts and MWS clients

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. In the scalability analysis of the Most Host, we have adapted BinXML [6] for compressing the MWS messages. BinXML format has reduced the size of messages by ~30%, thus hugely reducing the transmission delays of MWS invocation. Apart from these main QoS features, we also have studied the issues with mobility, failure and rejoining of mobile nodes and automatic start-up of the Mobile Hosts to save the smart phone resources. [11]

Similarly, huge number of available WS, 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. Discovery analysis of Mobile Host identified the *mobile P2P discovery* [11] mechanism that relied on existing Peer to Peer (P2P) networks for advertising and discovering the services. The study realized a solution in JXTA/JXME network utilizing the JXTA *modules* feature. The approach also considered categorizing the services and the advanced features like context aware service discovery. The evaluation of the discovery approach suggested that the smart phones are successful in identifying the services, with reasonable performance penalties for the Mobile Host. [12]

However, the QoS and discovery analysis of MWS 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) [14] is designed and established as an intermediary between the WS clients and the Mobile Hosts within the Mobile Enterprise. MWSMF is realized using ESB technology and open source ServiceMix [3]. Figure 1 shows the Mobile Enterprise, with the participating Mobile Hosts, clients, MWSMF and external application servers etc. While the MWSMF was successful in achieving the integrational requirements of Mobile Host and Mobile Enterprise, a standalone framework again faces the troubles with heavy loads. Hence to scale the system on demand and to provide the load balancing for the Mobile Enterprise, we are turning to the cloud computing paradigm.

III. SCALABLE MOBILE ENTERPRISE

The problems with scalability and elasticity for the Mobile Enterprise are quite relevant and the system should scale on demand to the loads of cellular enterprise. For example number of Mobile Hosts providing the services and the number of services provided by each Mobile Host can explode while some events are underway; like Olympics or national elections etc. Some of these application scenarios are addressed in [11]. This increases the number of clients the framework 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. However, the cloud based load balancing for the Mobile Enterprise can be at the framework level or at the individual service level. We try to address both the approaches in terms of two application scenarios, Remote patient tele-monitoring scenario and lightweight application server (LAS) services for mobiles.

In a remote patient tele-monitoring scenario, the Mobile Host can collect vital signs like blood pressure, heart rate, and temperature etc, of a patient who is located remotely, from different sensors and provide them to the doctors in real time. In the absence of such Mobile Host the details are to be regularly updated to a server, where from the doctor can access the details. The latter scenario causes problems with stale details and increased network loads. [12]

A. Scalable Remote Patient Tele-monitoring System

In the Remote Patient Tele-monitoring System, the main load for the Mobile Enterprise was at handling large number

of patients and their doctors, along with providing the QoS services from the MWSMF. For example the MWSMF has to convert the incoming XML based messages to BinXML format so that the messages can be exchanged across the radio link [14]. The process is taken care by the QoSVerifier component of the MWSMF (figure 1). So to increase the elasticity for the Mobile Enterprise we can establish a load balancer in the cloud and establish MWSMF on several nodes in the public cloud, handling the mobile clients by accessing services from several Mobile Hosts.

To test the scenario, the MWSMF was realized on the EC2 cloud. Once the 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. There are several load balancing techniques that can be used in this scenario. One approach is to use Domain Name System (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. 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. Another approach is to use load balancing proxy server in front of MWSMF nodes. Among several options, Apache HTTPD server with mod_proxy and mod_load_balancer is probably most commonly used configuration [2, 16]. 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. Alternative http proxy load balancer HAProxy [15] allows such dynamic behavior and it was considered in our second scenario.

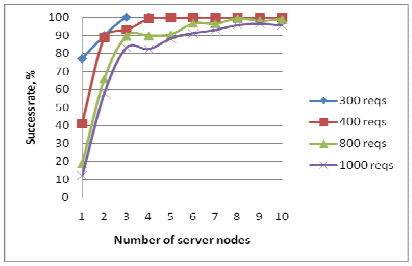

Figure 2. Success rate of MWSMF requests over concurrent requests

The tests are conducted with a load balancer and a set of worker MWSMF nodes established in EC2 cloud infrastructure with 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 [1]. 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 [2]. Load balancer was setup to use request based scheduling, which means that all worker nodes received equal amount of requests. The tests were also critical in analyzing the scalability of the MWSMF.

To test the scenario, we established a JMeter based client network that sent 5 consequent requests of 300 to 1000 concurrency requests level on the load balancer. This simulates a large number of simultaneous clients for the MWSMF and the Mobile Host in Mobile Enterprise. 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 succeed. The results of the experiment are shown in figure 2. From the diagram it can be observed 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. In summary we observed that, with current MWSMF implementation one single node can handle around 100-130 concurrent MWS requests with 100% success rate and adding an additional node adds around 100 new concurrent requests to the total capability. However for loads greater than 800 concurrent requests, the load balancer itself becomes the bottleneck and adding any additional nodes do not give desired effect. This showed mediation framework to be horizontally scalable and elastic cloud environment helps to achieve required setup very quickly.

B. Mobile access to LAS services

Lightweight application server (LAS) is designed as a community middleware that is capable of managing users and multiple hierarchically structured communities. It also supports access rights to a set of services. LAS mainly offers MPEG-7 (Moving Picture Experts Group) multimedia services. MPEG-7 is a well-established and widely used standard in multimedia data management. However, due to its inherent complexity it was not used in mobile data management that often. With new initiatives like the application profiles, the use of MPEG-7 has become much easier, also for mobile data management. A community application can make use of the offered services by simply connecting to the server and then remotely invoking service methods. Server functionality of the LAS is easily extensible by implementing and plugging in new services and respective components. Many community information systems have been built on top of this framework including MIST; a MPEG-7 based non-linear digital storytelling system, ACIS; a multimedia information system, and CAAS; a mobile application for context-aware search and retrieval of multimedia and community members. [10]

Even though, LAS is a reliable application server, it is not a pure WS architecture; it was not designed under the SOA paradigm and important aspects like scalability and distributed services were not taken into account. QoS and performance problems have been observed recently by LAS users. Moreover, the multimedia services are now also being offered to the Mobile Hosts and mobile phone users. Generally LAS services are extended and are provided as services from the Mobile Host. The extensions can be user specific. This entry of the Mobile Hosts into the LAS has further increased the scalability problem. Load balancing and cluster support were observed to be the immediate requirements for improving the performance of the LAS.

C. *Load balancing mobile access to LAS services*

Contrary to the Remote Patient Tele-monitoring System, the QoS of the LAS can be improved either by changing the architecture of the LAS to have the cluster support or to employ a binding component on the MWSMF, taking care of the load balancing issues. In the first case, a hardware based load balancer can be deployed through specialized devices like multilayer switches [16]. However, implementing, configuring and maintaining this solution is costly and time intense. Alternatively, we can deploy the LAS servers on the cloud and employ the same load balancer technique as in the case of the first scenario. However current LAS are standard servers with sufficient resources. We had limited access to these servers and services; and we decided not to change existing setup. As a third solution, we deployed the HAProxy node in the cloud and the requests are diverted to the respective LAS. If the load further increases, new LAS nodes can be deployed on the cloud. The performance latencies for this solution are similar to that of the MWSMF. The main difference is that HAProxy allows dynamic behavior to the architecture. New LAS nodes can be added dynamically to the setup. This solution utilizes the elasticity, dynamic and on-demand provisioning features of the cloud, to the most.

In the second option; employing a binding component on the MWSMF, we developed components that provide WS interface to the LAS services. The components are designed so that the requests are diverted to the less occupied server among a set of LASs. Thus load balancing is handled by the LASBalancer component at MWSMF. Communication from LASBalancer to the LAS is according to LAS requirements. Inside LAS there are no necessary changes to do. Mobile users of LAS only need to connect to a single point, the MWSMF, in order to access any LAS server they are interested in. Without this solution, Mobile Hosts should have specific connection to the right LAS server based on the services offered by it. However, this architecture adds extra load to the mediation framework at LASBalancer level. QoS and fault tolerance features of ESB help to some extent, in handling this load. But, LAS requests don't need QoS transformation features of the MWSMF as the messages are sent via Internet. So the node that provided load balancing and WS interface for the LAS, is separated from the MWSMF, and we deployed it on the EC2 cloud. The HttpInvoker just diverts the LAS requests to this node. Now this node is horizontally scalable and we can apply business logic, fault tolerance and solution correctness to the cloud node without gravely affecting the performance of MWSMF.

IV. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

The developments in the web services domain, the improved smart phone capabilities and the enhanced transmission capabilities of the cellular networks have lead to the MWSP domain. The paper summarized the challenges and research associated with 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 have raised the necessity for a middleware framework. The features, realization details and the scalability issues of the MWSMF are discussed.

However to scale of Mobile Enterprise to the loads possible in mobile networks, we tried to shift some of its components to the cloud computing paradigm. The paper tried to illustrate this categorical shift in terms of two application scenarios. It showed that MWSMF is horizontally scalable, thus allowing to utilize cloud's elasticity to meet load requirements in an easy and quick manner. It also illustrated different means to scale the LAS based MPEG-7 services. Thus cloud computing is shown to scale the Mobile Enterprise dynamically. Our future research in this domain will focus at surge computing and auto scaling so that Mobile Enterprise can scale according to the oscillating loads automatically.

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 research cluster, RWTH Aachen University.

REFERENCES

[1] Amazon Inc, "Amazon Elastic Compute Cloud (Amazon EC2)", http://aws.amazon.com/ec2/ [* URLs last visited on 29th Apr 2010]

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

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

[4] M. Armbrust et al., "Above the Clouds, A Berkeley View of Cloud Computing", Technical Report, University of California, Feb, 2009.

[5] 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.

[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] Google Inc, "App Engine", http://code.google.com/appengine/

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

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

[10] M. Spaniol, R. Klamma, H. Janßen, D. Renzel, "LAS: A Lightweight Application Server for MPEG-7 Services in Community Engines", Int. Conf. on Knowledge Management (I-KNOW '06), pp.592, 2006.

[11] S. N. Srirama, "Mobile Hosts in enterprise service integration", PhD thesis, RWTH Aachen University, 2008.

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

[13] S. N. Srirama, M. Jarke, and W. Prinz, "Mobile web service provisioning", In Int. Conference on Internet and Web Applications and Services (ICIW 2006), page 120. IEEE Computer Society, 2006.

[14] S. N. Srirama, M. Jarke, and W. Prinz, "MWSMF: A Mediation Framework Realizing Scalable Mobile Web Service Provisioning", In Int. Conf. on MOBILe Wireless MiddleWARE, Operating Systems, and Applications (Mobilware 2008), ACM.

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

[16] W. Tarreau, "Making Applications Scalable with Load Balancing, Revision 1.0", Sep, 2006. http://1wt.eu/articles/2006_lb/