

# Performance Comparison of a SOAP and REST Mobile Web Server

Anonymous ICOSST-2009 submission

Paper ID 23

## Abstract

*Although processing power and memory storage are no more issues in modern electronic devices, but a system software must always be light enough to respond fast in order to meet real-time constraints. This is perhaps an ultimate necessity of resource-constrained devices. Considering a Mobile Web Server (MWS) as a service delivery platform that needs to process hundreds of service creation requests, a performance evaluation of such system prior to deploying plays a key role in achieving a high-end solution.*

*This paper presents a performance comparison between the SOAP and REST MWS. A detailed comparison of service creation process in both mechanisms has been done with the help of state-of-the-art queuing theory and a proposed mathematical model. The comparison is studied based on the performance metrics such as server utilization, request waiting-time, throughput and average queue-length. Consequently, the REST is recommended as an underlying messaging framework for resource-constrained mobile devices.*

## 1. Introduction

As a service delivery platform, a MWS has to process hundreds of client requests daily such as service creation, subscription and billing etc. From the client-side view, these operations are to be provided at an acceptable Quality of Service (QOS) level. Consequently, the architectural performance analysis of the MWSs is required prior to deployment in terms of server utilization, throughput and request waiting-time for different service payloads and request arrival rates.

SOAP is widely used as a messaging framework for web services realization of Service Oriented Architecture (SOA). This standard is not suitable for resource-constrained mobile devices, provisioning Mobile Web Services, due to large payloads of SOAP message constructs. Apart from memory limitations, the data parsing of the intensive SOAP messages could lead to performance degradation of the MWS in terms of server utilization. In con-

trast, the REST messaging framework is light enough to be appropriate for messaging communication in such devices. Each service resource in this framework is identified by a single URL only.

Several performance analysis methodologies have been used in past for performance evaluation of web servers. An open queuing network model was used for web server modeling in [7]. In [6], a performance model of an asynchronous web server is presented which is validated by CSIM simulation language. Hernandez-Orallio and Vila-Carbo, in [5], proposed a statistical model for web traffic of a single-site web server based on the histogram analysis. Moreover, the model provided is also validated with real-time workload traces. Moreover in [8], the M/G/1/K queuing model is used for performance modeling of a web server.

In this paper, we present a performance comparison of the Mobile SOAP Server (MSS) and the Mobile REST Server (MRS) of the MWS presented in [1][2]. As a model based analysis, the service creation process is focused to study the performance of the MSS and the MRS with underlying synchronous and asynchronous interaction system. The rest of the paper is organized as follows: Section 2 defines the service creation process of synchronous and asynchronous mobile web services. Section 3 presents the system model used to compute the request waiting-time in the MWS. Section 4 describes the performance metrics used for performance evaluation. In section 5, an extensive comparison of MRS and MSS is presented based on the results obtained from the real-time measurements, queuing theory simulator and the proposed analytical model described in the section 3. Finally, a conclusion is drawn in section 6.

## 2. Service Creation Process

This section describes the service creation process of synchronous and asynchronous mobile web services. Generally, apart from service type, a service creation process starts when the MWS receives a service creation request from the network and ends with the creation of requested service object. This means, the service execution is not considered as a part of service creation process. The messaging

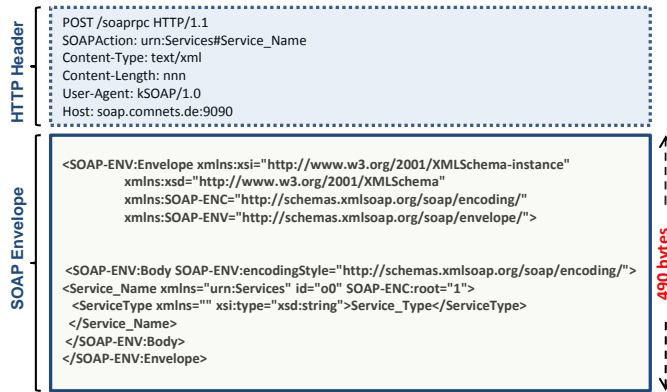


Figure 1. SOAP Service Creation Request

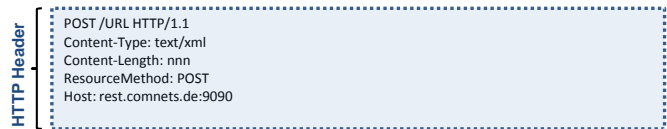


Figure 2. REST Service Creation Request

framework, such as SOAP or REST, does not effect the service creation process directly, that is, the steps of service creation process are same for each messaging framework. However, the time taken by the service creation process in both the cases differs based on the parsing overhead of their respective message constructs. The parsing of received request message is necessary in order to identify the name, type and context data of the service being created. Irrespective of underlying interaction style, the typical service creation requests with SOAP and REST messaging protocol are depicted in figures 1 and 2 based on [3][4]. The service is requested with no context data in these requests in order to calculate the size of payload only. In figure 1, a request message is comprised of the HTTP headers and the SOAP Envelope. The name of the service being created is identified by HTTP header's SOAPAction element, whereas a whole SOAP Envelope of 490 bytes is used only to identify the type of the service. The type of service can be synchronous or asynchronous in this case. On the other hand, in case of REST messaging example in figure 2, the request message is only comprised of HTTP header. Here, both the name and type of service are identified by the URL string.

### 2.1. Service Creation Process in synchronous interaction style

Synchronous interaction style is suitable to create short-lived services. These services are required to perform short-

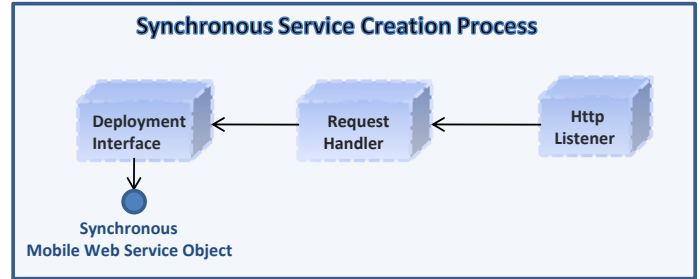


Figure 3. Synchronous Service Creation Process

lived tasks, for instance, a weather service is used to check the current weather status. During the entire process, the client PC is blocked in a waiting state.

Figure 3 depicts the service creation process of synchronous mobile web service in the MWS. When the server receives a service creation request through its HTTP Listener, it is delegated to the Request Handler. The Request Handler is responsible to parse the request and extract the name and context data of the requested service. The information acquired is then handed over to the Deployment interface for further processing. The Deployment interface maintains the list of all deployed services in a map table. Therefore, after receiving the service information from Request Handler, it checks the map table for the requested service and creates the service object in return.

### 2.2. Service Creation Process in asynchronous interaction style

In contrast to synchronous system, the asynchronous interaction is suitable for long-lived services. These services require a specialized mechanism for their control and management in order to facilitate runtime client interaction. This interaction is useful for feedback systems where intermediate results are collected from continuously executing services. For example in a health care system, a patient's blood pressure sensing service could be monitored by the doctor continuously or periodically. Similar to synchronous, in the asynchronous system the service creation process finishes when a service object is created. However, asynchronous system facilitates to start the service immediately or at some later time. Thus the concept of service invocation in the asynchronous systems is different from the service creation. The service invocation process is time dependent which usually starts after a certain predefined timer expires while the service is ready and ends with the completion of service task.

Figure 3 depicts the service creation process of an asynchronous mobile web service in the MWS. Similar to synchronous service request, an asynchronous service request is first delegated to Request Handler by HTTP Listener,

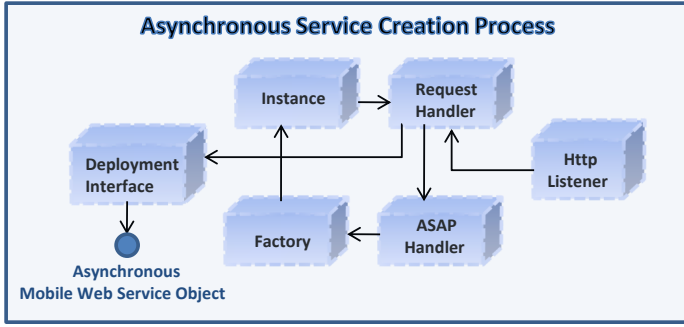


Figure 4. Asynchronous Service CreationProcess

which in return delivers the service name and context data to the ASAP Handler. The ASAP Handler delivers the control to the Factory which maintains the list of all services. The Factory creates a new instance of the requested service. The instance creates a new object of the RequestHandler which in turn sends the service information to the Deployment interface. The Deployment interface then either starts the respective asynchronous web service immediately or just assigns the asynchronous web service object to the Instance based on the information acquired from the service creation request.

### 3. Mobile Web Server Modeling

This section presents the system model used to estimate the waiting time of a service creation request in MSS and MRS of the MWS. Assuming MSS and MRS as queuing servers, a non-probablistic mathematical model used for waiting time estimation is presented, however the derivation of mathematical model is out of scope of this work, which will be published in another article. Following basic assumptions are made for the system modeling.

- **Single Server Queuing Model:** Based on different mechanisms of the service creation process in the MRS and the MSS with Synchronous and Asynchronous interaction, the MWS is modeled with four different single server models termed as Synchronous MSS (Syn-MSS), Asynchronous MSS (Asyn-MSS), Synchronous MRS (Syn-MRS), Asynchronous MRS (Asyn-MRS). Every individual sever model has an infinite queuing capacity. The requests are assumed to arrive in queue in a First Come First Serve (FCFS) manner.
- **Poisson Arrival Process:** In each server model, the service creation requests arrive in a poisson manner with rate  $\lambda_m^k$  requests per second, where  $m$  represents the messaging framework such as SOAP or REST and  $k$  represents the kind of interaction system such as synchronous or asynchronous.

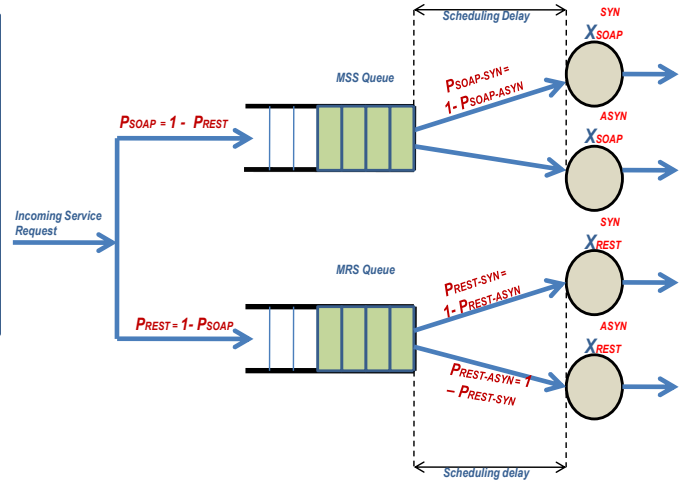


Figure 5. Web Server Model

- **Deterministic Service Time:** The service time is the time taken by an individual server to complete its service creation process. Upon the arrival of every request, the corresponding queuing server has to perform same action, which requires the same time to complete. Therefore, the service time distribution is considered as deterministic with constant time,  $D_m^k$  seconds.
- **Mean Scheduling delay:** Scheduling delay is the time taken by a request from leaving the queue till it reaches the server. This time is based on the underlying scheduling mechanism of the system software. Therefore, it might not be same for each request, however; we assume that a mean value is known in order to study the system behaviour.

The fig 5 shows the queuing model of MWS. An incoming service creation request is equally probable to enter either in the MSS or MRS queue. Similarly, the probabilities that a request in either queue will enter the corresponding synchronous or asynchronous server are equal. The model is comprised of four single server queuing models i.e., M/D/Syn-MSS(1), M/D/Asyn-MSS(1), M/D/Syn-MRS(1), M/D/Asyn-MRS(1). Although the model is capable of studying the performance of the MWS as a multiple server model with varying service times, the main focus of this paper is to compare the performance of MSS and MRS. Therefore, within the scope of this paper, we will consider the four single server queuing models one by one.

In order to study the single server models individually, the corresponding arrival probabilities will be assumed to 1. For example, considering M/D/Syn-MSS model, the corresponding  $P(SOAP)$  and  $P(Syn-SOAP)$  will be assumed to

1 while the rest of the probabilities will become 0, which are P(Asyn-SOAP), P(REST), P(Syn-REST) and P(Asyn-REST). Same will be the case while considering the rest of the single server models.

Based on the system modeling, the mathematical model for waiting time estimation in each single server model is given as:

$$W(\gamma(i)) = \eta(i) * \xi + [\delta(i) + (\eta(i) - 1) * D_m^k(S)] \dots (1)$$

where:

$\gamma(i)$  is the  $i$ th request arriving in the queue.

$\eta(i)$  is the position of  $i$ th request in the queue.

$\delta(i)$  is the remaining service time of a request in server.

$\xi$  is the mean scheduling delay of underlying system software.

$D_m^k(S)$  is the time of service creation process in a particular single server model.

## 4. Performance metrics

Here we describe the performance metrics used to evaluate and compare the performance of MSS and MRS.

- **Server Utilization:** Server Utilization is the percentage of time during which the server is busy processing jobs. It is computed as the ratio of the request arrival rate and the service rate. For a stable system, the value of this ratio must be less than 1 (100 %), that is, the request arrival rate should not exceed the service rate.
- **Average Waiting Time:** This is the time an arriving request has to wait in a queue until the server gets free and starts processing it.
- **Average Throughput:** Throughput is the number of the service creation requests processed per unit time. It should not be confused with the service time, as the service time does not include the waiting time while it does.
- **Average Queue Length:** It is the average number of the service creation requests in queue at a certain time.

## 5. Evaluation

Based on the metrics defined in the section 4, an extensive comparison of MSS and MRS is studied in this section. These metrics are evaluated by running the MWS in Java Micro Edition (Java ME) emulator. The results obtained are then validated with analytical model described in section 3 and state-of-the-art queuing theory. Queuing theory results are computed with a well-known queuing theory simulator, called QtsPlus.

The evaluation is done in two phases. The first phase focuses on server utilization in each single server model by

varying the request arrival rate and service context data. Whereas the second phase is devoted for the evaluation of rest of the performance metrics based on full server utilization obtained in the first phase.

### 5.1. Phase-I

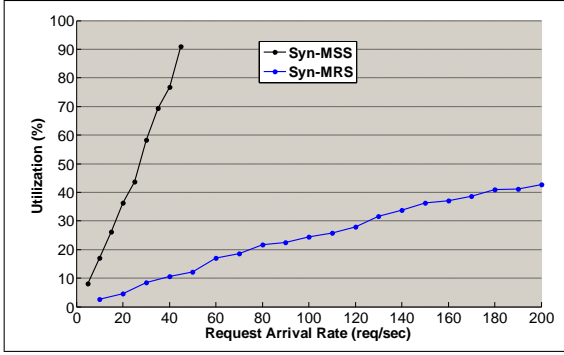
In this phase we will see the behaviour of MSS and MRS in terms of server utilization based on increasing request arrival rate and service context data. Due to limitations of our analytical model, the server utilization is only computed with QtsPlus in this phase.

Figure 6(a) shows the comparison of the Syn-MSS and the Syn-MRS single server models in terms of server utilization. It is shown that the Syn-MSS is almost 90% utilized with the arrival rate 45 req/sec, whereas the Syn-MRS is approx. 10 % utilized with the same arrival rate. The figure shows that 45 arrival rate is the maximum rate that the Syn-MSS can serve in a stable state, while the Syn-MRS is only 42.8% utilized with even 200 arrival rate. By seeing the behaviour of the Syn-MRS, one can easily figure out that it will touch the boundary of atleast 400 arrival rate upto its full capacity as the service time is constant for each request.

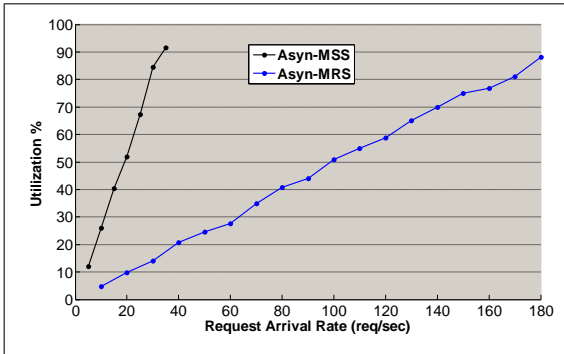
Similarly comparing the Asyn-MSS and the Asyn-MRS in figure 6(b), the Asyn-MSS is fully utilized at the arrival rate 35 req/sec, while the Asyn-MRS is less than 25% utilized with the same arrival rate. Here, the figure shows that Asyn-MRS is almost fully utilized at arrival rate 180 req/sec.

Figure 7, depicts that how service time is dependent on the size of service context data in both server models and how it effects the server utilization. It is worth noting in figures 7(a) and 7(b) that in the Syn-MRS, the service time and the server utilization are constant for every increase in size of service context data, that means, size of context data does not effect the performance of the Syn-MRS at all. This is because of no context data parsing is done in service creation process of the Syn-MRS. However on the other hand, the Syn-MSS service time and server utilization are increasing in steps with the increase of context data. Comparing the Asyn-MSS and the Asyn-MRS in figures 7(c) and 7(d), the behaviour of both the servers is same, however, the Asyn-MRS is more efficient with 51% utilization compared to 92% utilization of the Asyn-MSS with 100 context data elements.

The analysis all the results obtained in this phase, apart from messaging framework, server utilization is much better in underlying synchronous interaction system as compared to asynchronous system. This is because of the service creation request in the synchronous system is lighter than the asynchronous system in terms of mandatory payload data needed to facilitate the service creation process.



(a) Synchronous System



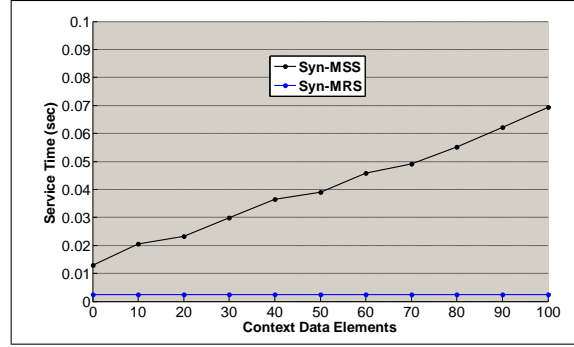
(b) Asynchronous System

Figure 6. Server Utilization Based on Variable Request Arrival Rates

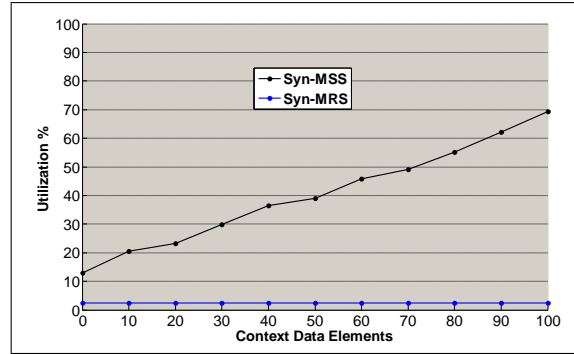
## 5.2. Phase-II

This phase shows an intensive comparison of the MSS and the MRS in terms of waiting-time, throughput and queue-length. For each single server model, analytical results are validated by real-time measurements and accuracy is compared with QtsPlus. The real-time measurements are taken by running the MWS in Java ME Emulator

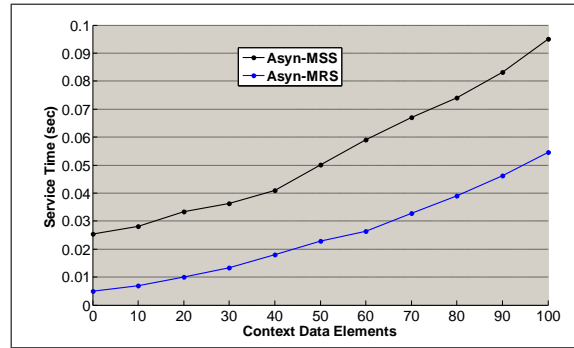
Figure 8 shows the waiting-time comparison of the MSS and the MRS based on varying request arrival rate. Considering the real-time measurements in figure 8(a) and 8(b), waiting-time is increasing in somewhat similar fashion in both server models, however, the mean waiting-time of a client request in the Syn-MSS is more than the Syn-MRS throughout the experiment. This is because the Syn-MSS requires more time to process one request as compared to the Syn-MRS as shown in figure 7(a). Here, the Syn-MSS takes 0.013 sec to serve a request with no context data, while the Syn-MRS takes only 0.002 sec to process same request, which is approx 84.66% less than the time taken by the Syn-MSS. Similarly, comparing the results obtained by the Asyn-MSS and the Asyn-MRS in figures 8(c) and 8(d) respectively, the overall behaviour of waiting-time curves of real-time measurements is similar, however, the



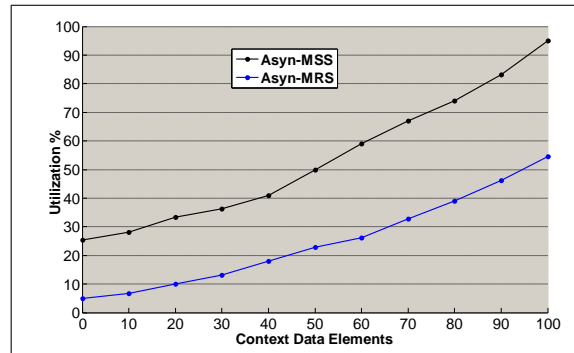
(a) Synchronous System



(b) Synchronous System



(c) Asynchronous System



(d) Asynchronous System

Figure 7. Server Utilization based on Variable Context Data Size

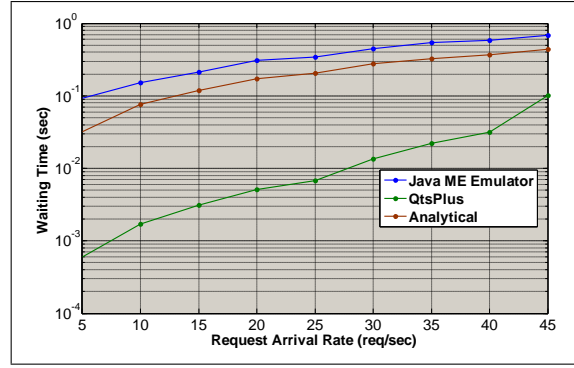
Asyn-MRS waiting-time response is better as compared to

the Asyn-MSS. The reason is even same in this case that the Asyn-MRS service time(0.005 sec) is about 80% less than that of the Asyn-MSS(0.025 sec) as shown in 7(c).

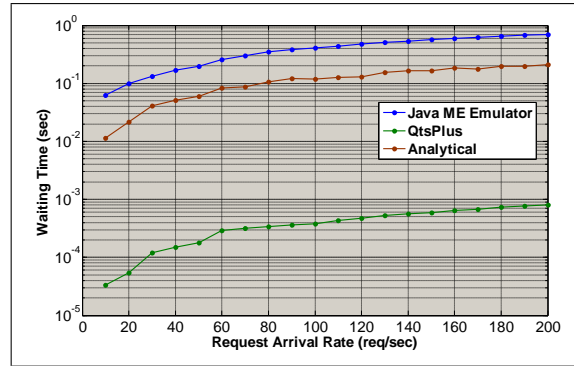
Apart from the real-time measurements, waiting-time results of our analytical model are also depicted in Figure 8 along with queuing theory results for each case. Although not same but the analytical results are quite comparable to the real-time measurements with similar behaviour. It can be seen that as compared to queuing theory results, the analytical results are more closer to the real-time measurements. One possible reason could be that our analytical model fairly caters the scheduling delay of underlying system-software when computing the mean waiting-time whereas queuing theory does not.

The throughput analysis of both the servers is depicted based on the variable arrival rate in figure 9. The figure shows the decreasing behaviour of the system throughput in each server model with increase in the request arrival rate. By increasing the arrival rate the waiting time of a request increases, which results in decrease of the system throughput. Comparing figures 9(a) and 9(b), at 45 arrival rate the Syn-MSS can process about 1.41 requests per second, whereas the Syn-MRS serves about 5.40 requests with the same arrival rate, which is approx 73% greater than that of the Syn-MSS. Here, the throughput behaviour of the analytical model is more identical to the real-time measurements as compared to queuing theory. Similar to the synchronous, the behaviour of both the servers in the asynchronous interaction is similar, however, the system throughput is better in the MRS as compared to the MSS. Here at 35 arrival rate, the Asyn-MRS throughput performance is approx 66% (4.5) better than that of the Asyn-MSS(1.51).

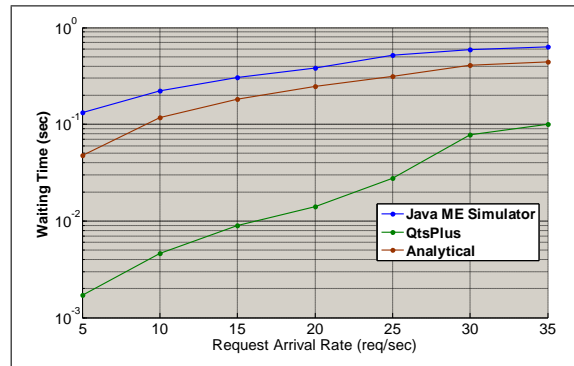
Figure 10 depicts the average queue-length comparison of both the MSS and the MRS. Analysing the figures 10(a) and 10(a), one can easily figure out that queue-length is rapidly increasing in the Syn-MSS as compared to the Syn-MRS. At 45 arrival rate, the Syn-MSS queue length is increased till about 30.95 requests, whereas it is only about 8.5 in the Syn-MRS with the same arrival rate. This means similar to the throughput analysis, the Syn-MRS performance is approx 73% better than the Syn-MSS, with about its maximum arrival rate, in terms of the average queue-length. With reference to the Asyn-MSS and the Asyn-MRS server models the figures 10(a) and 10(a) depict that the average queue-length in both the models is increasing in a bit similar fashion, however, the average queue-length in the Asyn-MSS is more than that of the Asyn-MRS with the same request arrival rate. Comparing the real-time measurements at the arrival rate 35 req/sec, the queue in the Asyn-MSS is about 72% more occupied with requests as compared to the Asyn-MRS.



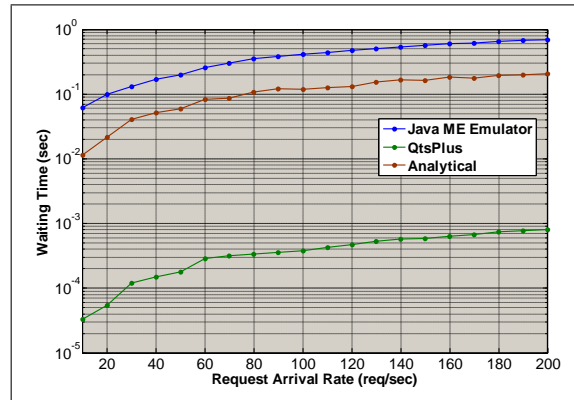
(a) Syn-MSS



(b) Syn-MRS

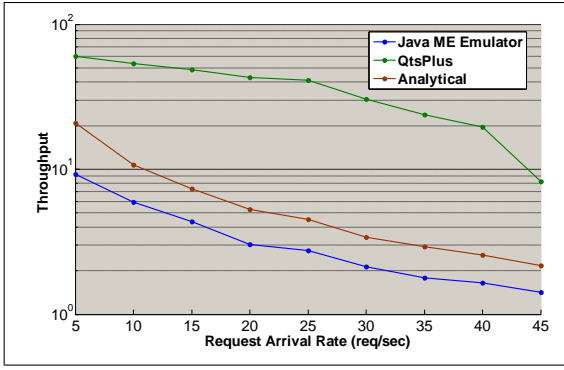


(c) Asyn-MSS

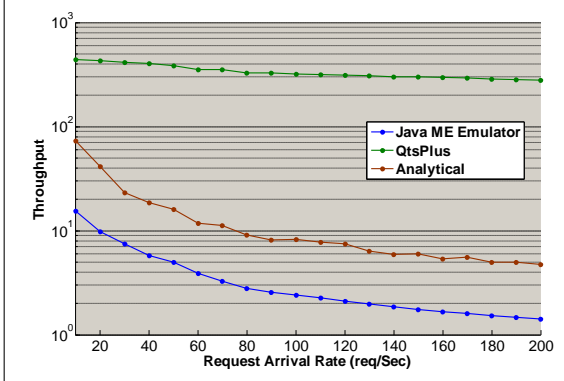


(d) Asyn-MRS

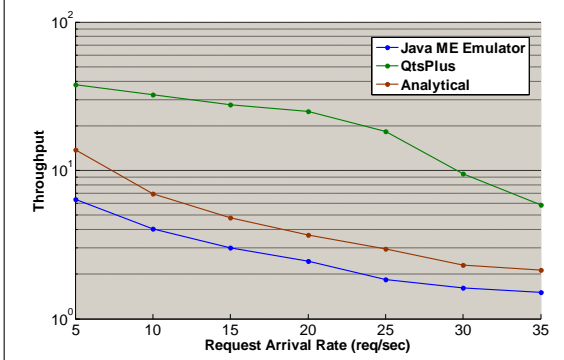
Figure 8. Waiting-Times based on Variable Request Arrival Rates



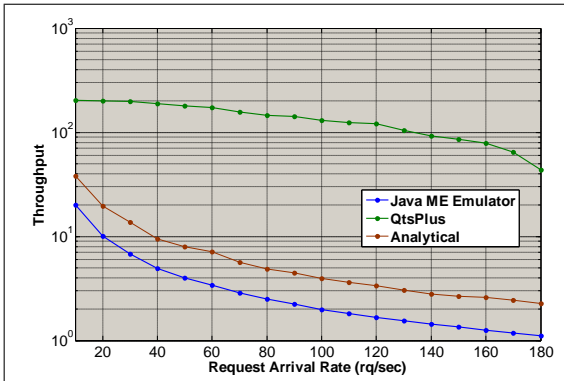
(a) Syn-MSS



(b) Syn-MRS

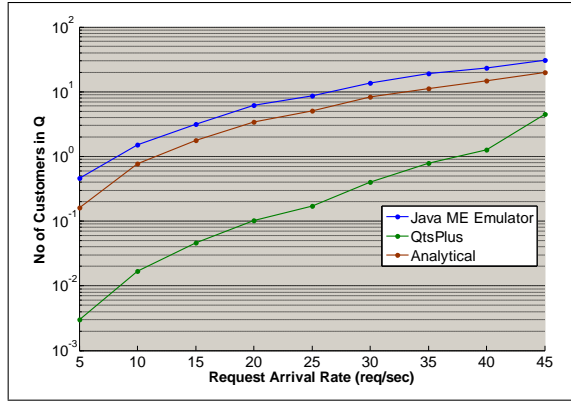


(c) Asyn-MSS

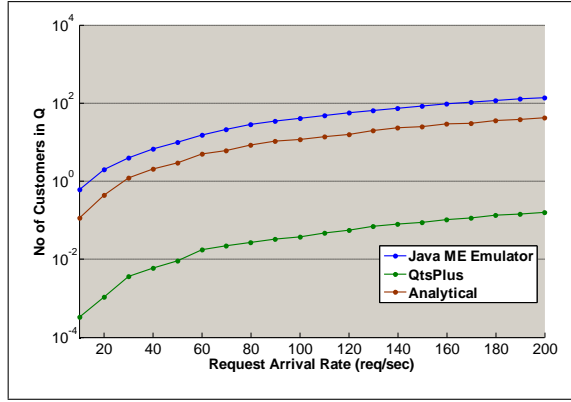


(d) Asyn-MRS

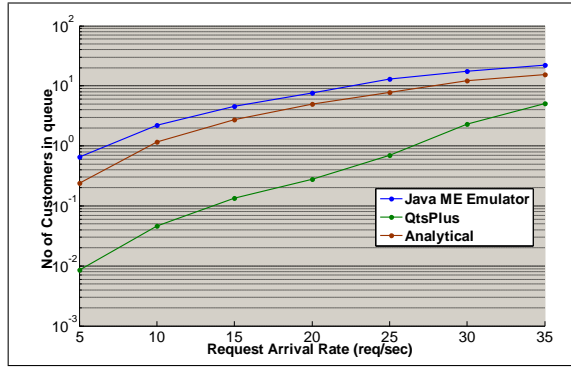
Figure 9. Throughput based on Variable Request Arrival Rates



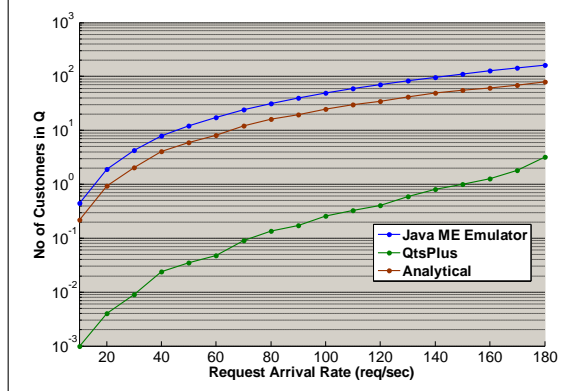
(a) Syn-MSS



(b) Syn-MRS



(c) Asyn-MSS



(d) Asyn-MRS

Figure 10. Queue-Length based on Variable Request Arrival Rates

## 6. Conclusion

The paper focuses on the architectural performance analysis of the MWS for the SOAP and REST messaging frameworks. An extensive comparison of the service creation process in the both mechanisms has been studied by employing the state-of-the-art queuing theory and the proposed analytical model. The performance is evaluated based on the metrics such as server utilization, waiting-time, throughput and the queue-length. The study shows that the performance of the MRS is better than the MSS for each computed metrics. Therefore, the paper is concluded by recommending the REST as an underlying messaging framework for mobile web services communication in resource-constrained mobile devices.

## 7. References

- [1] Aijaz, F, Hameed, B. and Walke, B. Asynchronous Mobile Web Services: Concept and Architecture. In Proceedings of IEEE 8th International Conference on Computer and Information Technology. 07/2008
- [2] Guido G., Mobile Web Services Concepts, Prototype, and Traffic Performance Analysis, Dissertation, ABMT 54, 1. Auflage 2007
- [3] Aijaz, F, Syed, Z, Chaudhary, M. and Walke, B. Enabling High Performance Mobile Web Services Provisioning. Accepted for IEEE 70th Vehicular Technology Conference VTC Fall 2009.
- [4] Aijaz, F, Syed, Z, Chaudhary, M. and Walke, B. Enabling Resource-oriented Mobile Web Server for Short-Lived P2P Services. Accepted for 9th Malaysia International Conference on Communications (MICC'09).
- [5] Hernandez-Orrallo, Villa-Carbo. Web server performance analysis using histogram workload models. Computer Networks Journal 30th June 2009.
- [6] S.Gokhale, Gokhale, A. and Gray, J. Performance Analysis of an Asynchronous Web Server. In proceedings of COMPSAC (2) 2006.
- [7] L. Slothouber, A model of web server performance. In proceeding of International World Wide Web Conference, 1996.