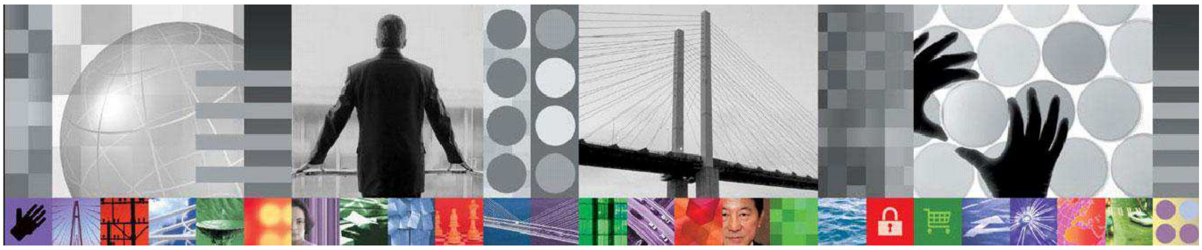
SIP and HTTP Converged Services Performance using the WebSphere Application Server

Oct 2009



Curtis Hrischuk, Ph.D.

1. Executive Summary

Enterprises are adding real-time communications capabilities into existing and new applications to help customers, partners and employees interact more efficiently. Optimizing user interactions can drive revenue while reducing costs: social networking can close sales, self-service is designed to help customers support themselves, and unified communications helps to speed-up business processes. Many enterprises start down the path of communications enablement by adding "Click to Call" capabilities to their existing applications. Adding "Click to Call" to an application enables users to click on an HTML button and start a voice conversation with a physical phone, a mobile phone or a Voice over IP (VoIP) softphone. Java^{TM} Enterprise Edition (Java EE) technologies make it easy to communications enable applications using Session Initiation Protocol (SIP) programming standards such as JSR 116 [JSR116] or JSR 289 [JSR289], as well as the IBM multi-modal development framework provided in the IBM® WebSphere® Application Server V7 Feature Pack for Communications Enabled Applications (CEA). WebSphere Application Server easily handles this cutting edge functionality by delivering over 3 million converged operations per hour, running on a single IBM BladeCenter® HS21 blade server.

This report is based on a benchmark for communications enabled applications that combines HTTP and SIP. The benchmark simulates people using a browser to establish a Voice over IP (VoIP) phone call and interact with a voice mail server. Note that the technologies utilized in this benchmark are not limited to VoIP phone calls, and could equally connect calls over a standard landline phone or mobile phone. The system configuration is a three-tier model. Tier one is the load driver that simulates the activities of Web users. Tier two comprises multiple application servers that receive the requests. Unlike most web applications where the third tier is databases, the third tier for this application consists of SIP user agents that simulate the end user, the voice mail server, and the application server.

This benchmark demonstrates the carrier-grade performance of IBM WebSphere Application Server 7.0.0.5. Its capacity is measured using a new metric of *Converged Operations per Hour*. This is the sustained hourly rate at which new users start and complete the web voice mail application. It begins with the initial HTTP request and stops with the call ending after an average duration of 60 seconds. Per user it includes an average of 2.5 HTTP GET requests and 28 SIP messages.

IBM WebSphere Application Server achieved 3,351,600 Converged Operations per Hour (931 converged operations per second) on an IBM BladeCenter HS21 Series x® blade server: 2 CPU, quad core machine, running at 3.33GHz with 16 GBytes of RAM. This was achieved with an average CPU utilization of 65%. This is equivalent to 7,218,830 Busy Hour Call Attempts (assuming each call is 13 SIP messages) while also processing 8,379,000 HTTP requests per hour, on a single machine running the same application. The benchmark results showcases the ability of WebSphere Application Server to provide the performance businesses demand, as well as fast development of multimodal applications utilizing frameworks such as the IBM WebSphere Application Server V7 Feature Pack for CEA [CEA]. Other IBM WebSphere products can further enhance these capabilities, such as IBM WebSphere Real Time for Linux® which can provide for more control over latency, or IBM WebSphere Virtual Enterprise that uses autonomic workload control mechanisms to manage overload or denial of service conditions.

2. Social Web Application Performance

WebSphere Application Server is the foundation to deliver next generation web applications that incorporate voice, instant messaging, and other ways to socially interact. Converged applications can be rapidly built using the IBM WebSphere Application Server V7 Feature Pack for CEA [CEA] or by utilizing HTML and SIP programming. The CEA feature pack enables developers to use Java, HTML and JavaScript to build communications enabled applications by using Web 2.0 widgets and services, adding capabilities such as click to call, co-browsing, and call notifications to web applications. Alternatively, developers with SIP programming skills would utilize the converged HTTP and SIP Servlet container in the IBM WebSphere Application Server to communications enable existing and new applications. This paper illustrates how the carrier-grade application performance of IBM WebSphere Application Server extends to new social applications that combine both HTTP and SIP for new business tools or processes. A converged technology application, that accesses voice mail over the web, is used as a new benchmark. The system configuration is a three-tier model with tier one executing the driver that emulates the activities of Web users. Tier two comprises multiple application servers that receive the requests and establishes a three way voice calls with the third tier: the end user, the voice mail server, and the application server.

3. The Web Voice Mail Application

The application's scenario is quite simple, converting an HTML button click into a voice conversation. The scenario begins with a sales person at a coffee shop wanting to access their voicemail. They access the corporate network, bring up their personal page, and then click on a button to access their voice mail. The button sends an HTTP request to establish a SIP based, voice three way call between: the voice mail system, the user's soft-phone on their netbook, and the WebSphere Application Server. The WebSphere Application Server manages the session start and end; the voice traffic is assumed to be carried on a separate path as is customary for SIP. When the user is done they either: (1) hang up the soft-phone (terminate the call by SIP) or (2) click a button on the web page (end the call using HTML). The benchmark roughly equalizes the termination types (i.e., 50% hang up and 50% terminate by the web page).

This scenario translates into many SIP messages and several HTTP requests, as shown in Figure 1. The HTTP requests are shown on the left of the application server and the SIP message exchanges are shown on the right. The soft phone and voice mail server behave according to the best practices in rfc3725.[1] The voice mail server and application server exchange events, similar to rfc3842,[2] so that it is known how many voice mail messages are pending for the user. To keep things simple, the benchmark assumes that there are always four pending voice mail messages when the user initially dials in. The average time the user interacts with the voice mail system (i.e., call hold time) is 60 seconds.

[1] rfc3725: Best Current Practices for Third Party Call Control (3pcc)

[2] rfc3842: Message Waiting Indication Event Package

From a workload perspective, each user generates 2 or 3 web page requests, as well as 28 SIP messages which is not an uncommon amount of SIP activity. An interesting aspect of the message flows in Figure 1 is that much of the SIP processing occurs in parallel between the various SIP test driver end points, which is a very good stress test.

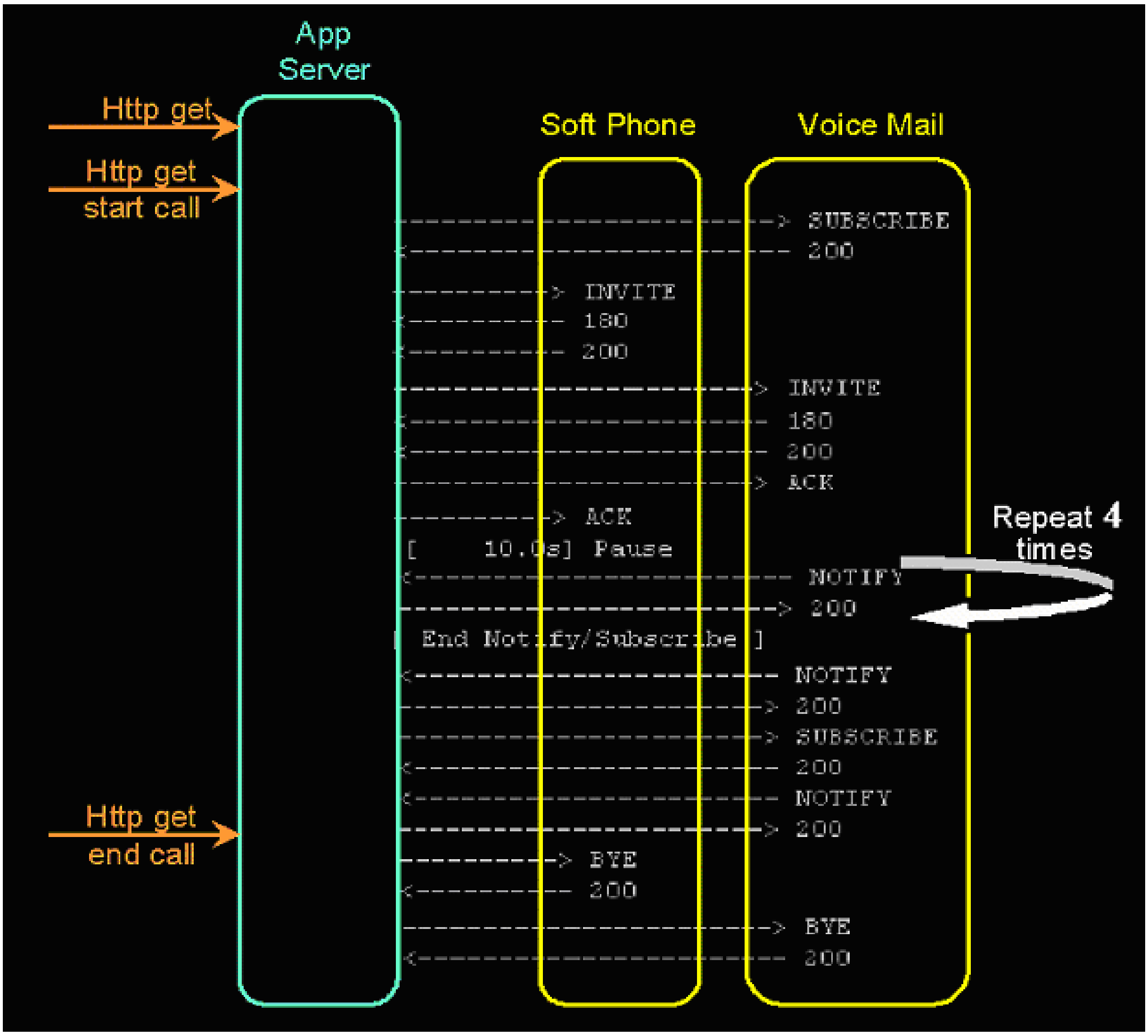


Figure 1: HTTP and SIP Message Exchanges

4. System Configuration

The benchmark configuration is shown in Figure 2. There are eight WebSphere Application Servers configured on the single hardware server. Running more than one application server is a standard technique of fully utilizing the hardware resources and increasing capacity (called *vertical scaling*). The application server hardware server is an IBM BladeCenter® HS21 Series x® blade server with a two 3.33GHz Intel® quad core CPUs with 16 GBytes of RAM, running Red Hat Enterprise Linux release 5.2.

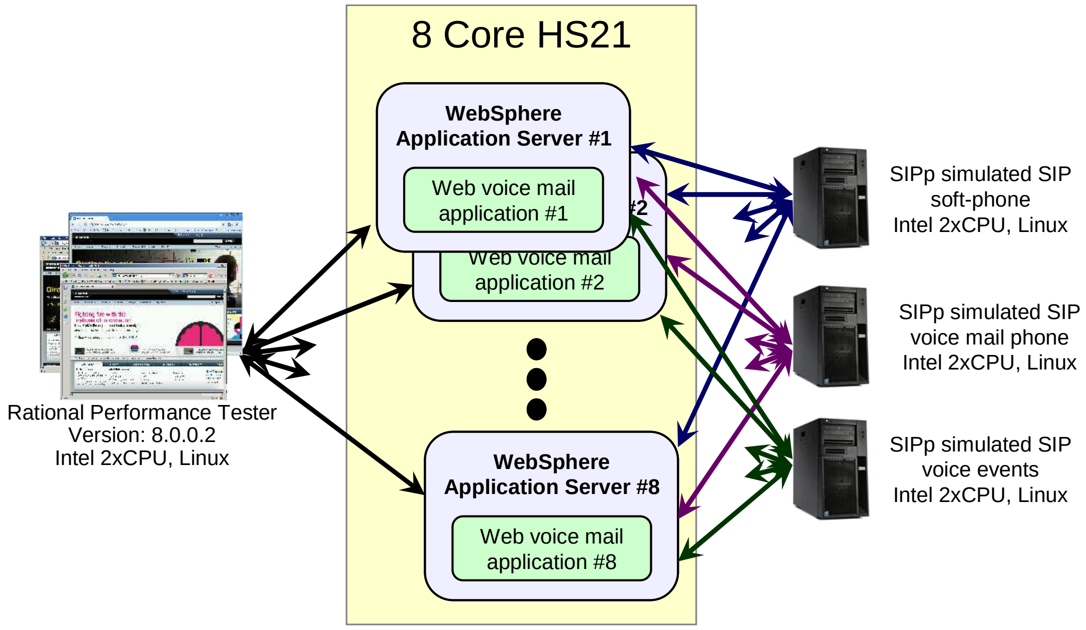

8 Core HS21
WebSphere Application Server #1
Web voice mail application #1
#2
Web voice mail application #2
WebSphere Application Server #8
Web voice mail application #8
Rational Performance Tester
Version: 8.0.0.2
Intel 2xCPU, Linux
SIPp simulated SIP soft-phone
Intel 2xCPU, Linux
SIPp simulated SIP voice mail phone
Intel 2xCPU, Linux
SIPp simulated SIP voice events
Intel 2xCPU, Linux

Figure 2: Software Configuration of a Single Hardware Server

IBM Rational® Performance Tester generates the HTTP load that simulates users accessing their voice mail over the web. This workload is generated using one workbench that manages sixteen load generation machines, each of which executes a single load driving agent. The application servers interact with the SIP load drivers that emulate the user's soft-phone and voice mail server. The traffic generator program SIPp [SIPp] is used to simulate the SIP end-points of Figure 1. There were three SIPp programs used: simulators of the user's soft-phone (two instances), the voice mail VoIP line (two instances), and a simulator of the voice mail server's events (four instances). Each SIPp program executed on its own machine with the exception of the SIPp event generation that had two programs per machine. This configuration ensured that the load generation mechanisms had ample capacity to drive the load so that the test harness capacity would not affect the measurements.

5. Performance Results

The performance results represent the combined throughput of the eight Java EE WebSphere Application Servers running on the single IBM BladeCenter® HS21 Series x® blade server.

The peak capacity is determined by increasing the rate of initial web voice mail HTTP requests until either the HTTP request timed out or one of the SIP interactions had an unexplained failure.[3] At that peak capacity the: user request rate, HTTP page request rate, SIP message rate,

[3] For the sake of efficiency, the SIPp traffic generator does not implement a SIP compliant protocol stack which can mislead SIPp into reporting an error occurred when it did not. This usually is the result of message retransmission or out of order messages being received, which follow the SIP protocol but may be falsely reported by SIPp to be an error. The SIPp XML script files do take into account some of these issues but, in some cases, manual inspection is required. SIPp does record enough information about each failure so that a manual inspection can determine if an error really occurred or SIPp was just confused.

CPU utilization, and aggregate used application memory are recorded. A Quality of Service latency value is also recorded, noting the delay between a SIP request and its acknowledgements ("95th Percentile latency from the NOTIFY to 200 acknowledgement message"). This delay is calculated from data recorded by the SIPp load driver, using a 15 minutes measurement interval at a fixed call rate.

The peak capacity of the web voice mail application is shown in Figure 3. Since the new benchmark combines both HTTP and SIP, a new capacity metric is used: *Converged Operations per Hour*.[4] This metric represents the sustained hourly rate at which new users start and complete the web voice mail application. It begins with the initial HTTP GET request and completes with the softphone hanging up or an HTTP GET that closes down the call. Per user it includes an average of 2.5 HTTP GET requests and 28 SIP messages.

The peak capacity is 3,351,600 Converged Operations per Hour (931 converged operations per second * 3600 seconds per hour).

The converged operations per hour metric can be broken down into the individual message rates driven by the workload. The HTTP request rate is 2,327 HTTP requests per second (roughly 931 converged operations per second * 2.5 HTTP requests per converged operation), while the SIP request rate is 26,068 SIP messages per second (931 converged operations per second * 28 SIP messages per converged operation). This combined throughput was achieved with an average 65% CPU utilization, allowing room for more advanced business logic without impacting the overall capacity. This is equivalent to 7,218,830 Busy Hour Call Attempts (assuming each call is 13 SIP messages) while also processing 8,379,000 HTTP requests per hour, on a single machine running the same application.

In addition to peak capacity, memory consumption is an important factor because users are logged in for long periods with this type of application. Although each user consumes a small amount of application memory, there can be many concurrent users so the application memory consumption can become a bottleneck. For example, there were about 55,860 users active at peak capacity (931 converged operations per hour * 60 seconds). The application memory that is used is part of the *Java heap.* Measuring the used Java heap across all application servers is a simple matter of finding the steady state Java heap memory used by each application server and then summing up the values.[5] As shown, the steady state memory consumption is 2,127 Mbytes total or about 266 Mbytes per application server. Since each application server was allocated a Java heap of 1,500 Mbytes this leaves roughly 83% of the application memory available. This memory measurement technique can also be used for comparison with other types of Java application servers that may split the SIP application processing across more than one tier.

[4] The term Converged Operations Per Hour is based on prior terminology. *Converged* is the usual industry term for web applications that use both SIP and HTTP. Telephony capacity is usually measured as Busy Hour Call Attempts. The new term joins these two concepts together.

[5] The IBM JVM's generational garbage collector is used in these measurements. The garbage collector activity is logged in the file native_stderr.log when enabled. The steady state memory usage is found by searching for the last 'global' garbage collection (i.e., 'gc type="global"') at peak capacity. Then the average used memory is calculated as the tenured 'totalbytes' value less the tenured 'freebytes' value. Other JVM vendors will record similar information.

An important quality factor for SIP is the round trip delay between issuing a SIP request and receiving the acknowledgement. If the acknowledgement is not received in a timely fashion, then the client retransmits the message. This window of time is not large compared with HTTP time-outs. For example, the initial message time-out value for a VoIP call is 500 milliseconds while the typical HTTP time-out value is 30 seconds: a factor of 60. Although the WebSphere Application Server detects and ignores retransmitted messages, the retransmissions use some processing power and also waste network resources, so minimizing retransmissions is beneficial. For this reason, the application round trip time is a useful quality measure because it provides an expectation of how close the system is to the retransmission threshold.

For the web voice mail benchmark, round trip SIP latency is measured as the interval from the voice mail server sending a NOTIFY and receiving the 200 OK acknowledgement. The 200 OK is generated by the application so measurement includes application latency; some other responses (e.g., 100 TRYING) do not reach the application but are returned by the lower level protocol processing. Figure 3 shows that 95% of round trip times are less than 60 milliseconds, 440 milliseconds below the retransmission threshold.

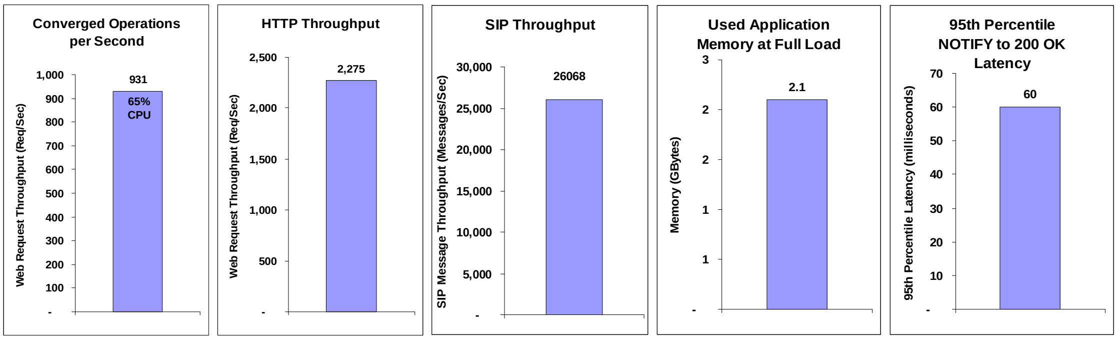


Figure 3: Performance of the Web Voice Mail Application

6. Conclusions

Enterprises are adding real-time communications capabilities into existing and new applications by combining the HTTP and the Session Initiation Protocol (SIP).

WebSphere Application Server is a leader in the industry for HTTP or SIP web application performance and this evaluation shows that the leadership of WebSphere Application Server extends to applications similar in scope to social networking, unified communications, self-serve, etc. web applications.

Although a single hardware server was used in this paper, the performance results can be extrapolated to additional servers because the WebSphere Application Server scales out horizontally to large deployments.

This high performance also extends into high availability applications. The built in high availability of WebSphere Application Server provides excellent capacity. WebSphere eXtreme Scale can expand upon this by providing a distributed, transactional, highly available cache which can be used for geographic redundancy. WebSphere eXtreme Scale could then be used to build high throughput transaction processing applications that may require web and/or voice multi-modal interactions.

Although not used in these measurements, there are other IBM WebSphere products to enhance these capabilities. If external factors require tightening up the latency values or having smarter management of retransmissions then WebSphere Real-Time for Linux can help to make the latency more deterministic and/or WebSphere Virtual Enterprise can use its autonomic admission controls to guard against extreme load or Denial of Service concerns.

7. References

[CEA] http://www-01.ibm.com/software/webservers/appserv/was/featurepacks/cea/features/

[JSR116] SIP Servlet API 1.0 Specification

[JSR289] JSR 289: SIP Servlet v1.1

[SIPp] http://sipp.sourceforge.net/

The information contained in this publication is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this publication or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

References in this publication to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth, savings or other results.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.