WORKLOAD SCHEDULER

VERSION 9.4

an IBM + HCL product

PERFORMANCE REPORT

Pier Fortunato Bottan
Giorgio Corsetti

Workload Scheduler
Performance Team - HCL Rome Lab

Document Version 1.0

This edition applies to version 9, release 4 of Workload Scheduler and to all subsequent releases and modifications until otherwise indicated in new editions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Workload Scheduler is a state-of-the-art production workload manager, designed to help customers meet their present and future data processing challenges. It enables systematic enterprise-wide workload processing for both calendar and event-based (real-time) workloads across applications and platforms. IBM Workload Scheduler simplifies systems management across distributed environments by integrating systems management functions. IBM Workload Scheduler plans, automates, and controls the processing of your enterprise's entire production workload . Pressures in today's data processing environment are making it increasingly difficult to maintain the same level of service to customers. Many installations find that their batch window is shrinking. More critical jobs must be finished before the workload for the following morning begins . Conversely, requirements for the integrated availability of online services during the traditional batch window put pressure on the resources available for processing the production workload.

IBM Workload Scheduler simplifies systems management across heterogeneous environments by integrating systems management functions.

1.1 What's New in Version 9.4

Workload Scheduler version 9.4 includes the following enhancements:

- Agent upgrade with minimal scheduling disruption
- Workload Scheduler Plug-in for IBM® Cloudant®
- Keeping track of changes to scheduling objects
- Auditing release management
- Version control
- Backup copy of tws_env script
- New event-driven workload automation action to open a ServiceNow incident
- IBM i job definition enhancements
- Passing variables between jobs
- Satisfying Requests for Enhancements (RFEs)

The auditing release management enhancement, in particular, was analyzed because of its potential impact on product performance.

For more details about Workload Scheduler version 9.4 enhancements, see the *Summary of enhancements* in the online product documentation in IBM Knowledge Center.

Dynamic Workload Console version 9.4 includes the following enhancements:

- Keeping track of changes to scheduling objects
- Auditing release management
- Version control
- Graphical view enhancements
- IBM i job definition enhancements
- Passing variables between jobs

The graphical view enhancement, in particular, was analyzed because of its potential impact on product performance.

For more details about Dynamic Workload Console version 9.4 enhancements, see the *Summary of enhancements* in the online product documentation in IBM Knowledge Center.

2. SCOPE

2.1 Executive Summary

The objective of this document is to report the performance results for the following new features delivered in Workload Scheduler V9.4.0.0, in addition to the previous performance improvements that have been consolidated in this release:

- Performance analysis of graphical view enhancements
- Performance analysis of auditing release management
- Scalability of concurrent users on the Dynamic Workload Console
- Performance analysis of the What-if analysis interactive Gantt chart

The main performance and scalability features verified in previous releases and, reported in the document, *"IBM Workload Scheduler Version 9.3.0.1 Performance and Capacity Planning Guide"*, have also been confirmed in release 9.4.0.0, specifically, the database plan status update throughput (mirroring), the dynamic agent schedule throughput and capacity planning guidelines. For the sake of practicality this document replicates most of the recommendations previously provided with some specific additional tuning options.

3. PERFORMANCE TEST

3.1 Test Approach

As specified in section 2.1, the majority of the performance tests were focused specifically on new features delivered in the 9.4.0.0 release. The guideline followed was to keep the performance benchmark results collected in previous releases and use them as key performance indicators (Table 1 outlines this benchmark) while new features are being adopted.

Scheduling throughputs, resource consumption and reliability are continuously certified assuring that there is no degradation with respect to the 9.3.0.x release. Specific tests have been implemented for the “What-if analysis” to provide information about some limitations that were detected and documented (code improvements are currently planned for the next fix pack).

Main Fault-tolerant Agent (FTA) – Dynamic Scheduling	• The plan includes **124,800 jobs scheduled during a 3-hour period**. In particular, there are **48,000 jobs** scheduled to start during a peak period from 11:00 to 11:10.
Workload Service Assurance	• 48 complex patterns, composed of multiple linked job streams (4) with 10 jobs each. 4 jobs for each complex pattern are defined as critical jobs. These additional 1,920 jobs are scheduled to start uniformly between 10:30 to 13:18.
Event-Driven Workload Automation	• **200 Workload Scheduler objects rules -** each rule matches a workstation and job name belonging to the daily production plan mentioned above and the success state of job execution. The action, in case of event matching, is to create a new message log. Normally, at the end of each test run, 4,140 events (Message loggers) are generated. • **File-created rules** -These event monitor rules generate a specific Message logger each time a new file with a predefined naming convention is created on each agent. In total, 240 events (Message loggers) have been generated each hour, that means 1 event every 4 minutes on each of the 16 agents. This kind of workload was planned to be turned on at 11:30 and to be turned off at 12:30.
Conditional Dependencies	• An additional workload (5%) of 3,200 jobs/800 job streams over 4 dynamic agents and 4 FTAs . This means that there are 100 job streams for each agent, half of which have internal dependencies and the other half external dependencies. These 100 job streams per agent are scheduled uniformly over time between 11:00 and 11:50. In the case with conditional dependencies, there are also 800 join conditions overall.
***Ad Hoc* Submission**	• Dynamic submission of jobs using the command "conman sbs" to submit a job stream with 5 different jobs with dependencies one from the others in a chain. In total, there are 1,000 dynamic jobs submitted over a period of 10 minutes. This dynamic job stream submission was planned to start between 12:40 and 12:50

Table 1. Daily plan workload composition

This workload is used as a standard benchmark for establishing key performance indicators whose baseline is continuously verified to track performance enhancements.

3.2 Environment

The test environment is based on LPAR nodes hosted on an IBM Power7® 8233-E8B (3GHz) server. All tests were performed in a 10 GB local area network. The LPAR has dedicated cores whose numbers were changed during benchmark executions.

The following table summarizes the software used and the version:

OS	AIX® 7.1 TL 03
RDBMS	IBM DB2® 10.5.0.8
J2EE	IBM WebSphere® Application Server 8.5.5.9 with SDK 8.0.2.10
LDAP	IBM Directory Server 6.3
Jazz™ for Service Management	JazzSM 1.1.3 with DASH 3.1.3.0
WS	9.4.0.0

Table 2. Software level of code

The HTTPS protocol was used and an IBM HTTP Server with IHS WebSphere Application Server Plugin acted as a load balancer with "Random" policy to distribute the user load on the Dynamic Workload Console servers. The procedure documented in the following link was followed to set up a high availability configuration (also known here as cluster):

http://www.ibm.com/support/knowledgecenter/SSGSPN_9.4.0/com.ibm.tivoli.itws.doc_9.4/distr/src_ad/ctip_config_ha_ovw.htm

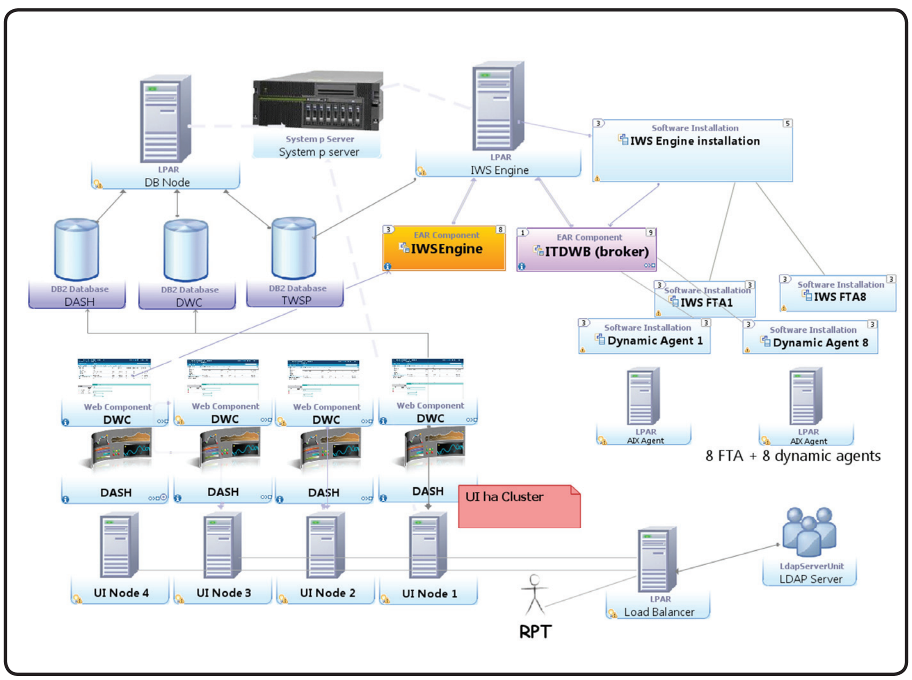

Figure 1. Overall deploy view of test environment

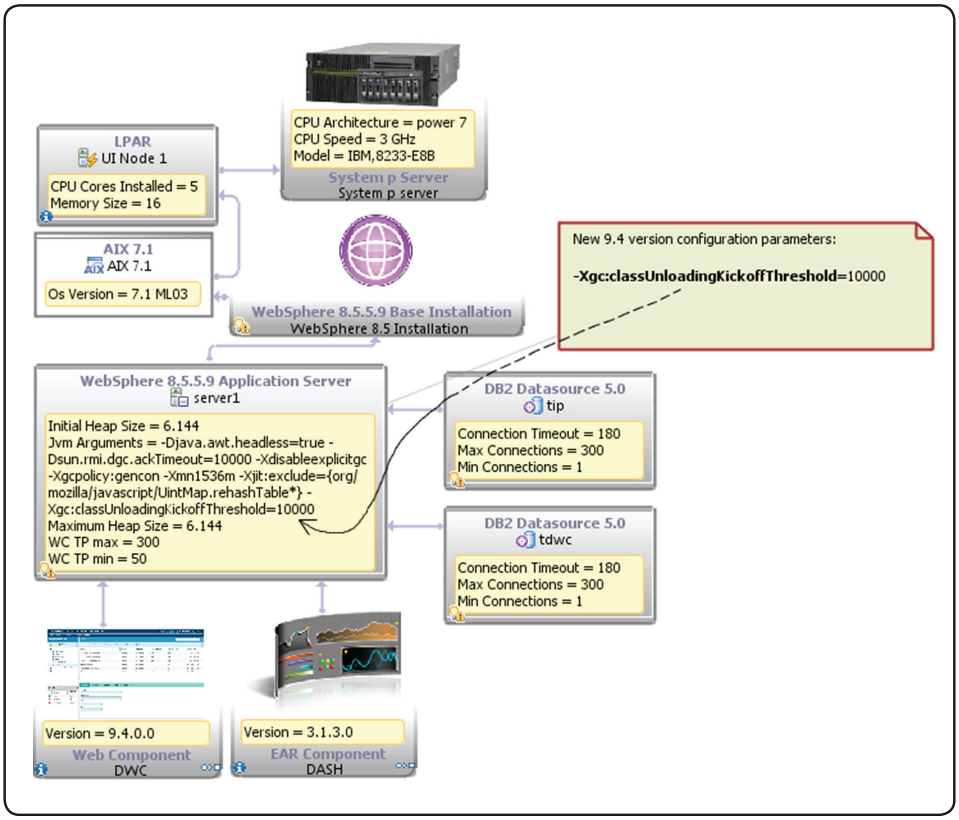

Figure 2. Dynamic Workload Console node configuration

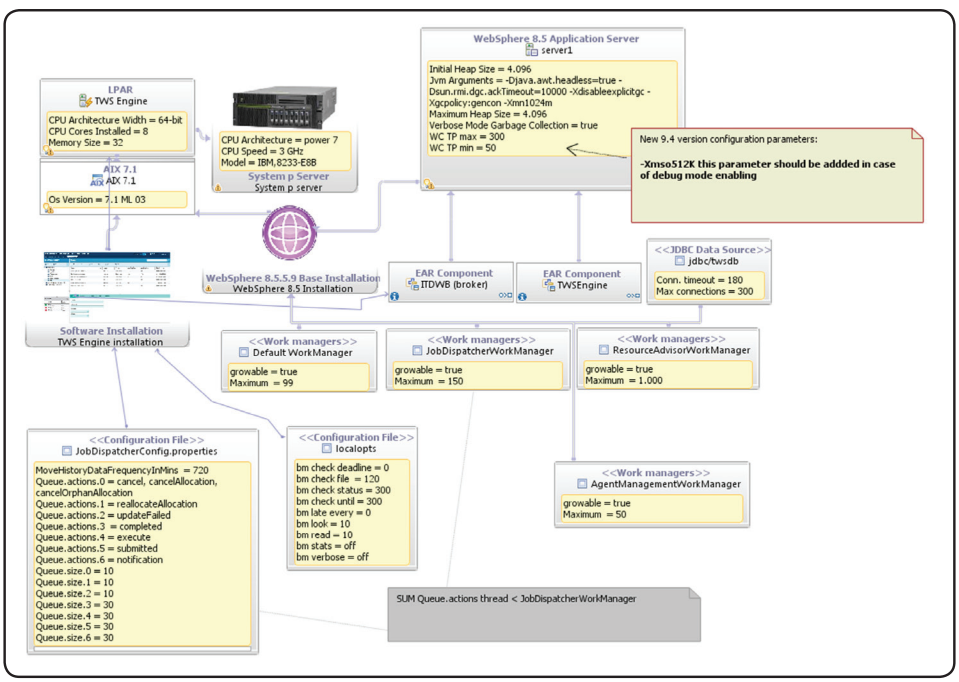

Figure 3. Engine node configuration

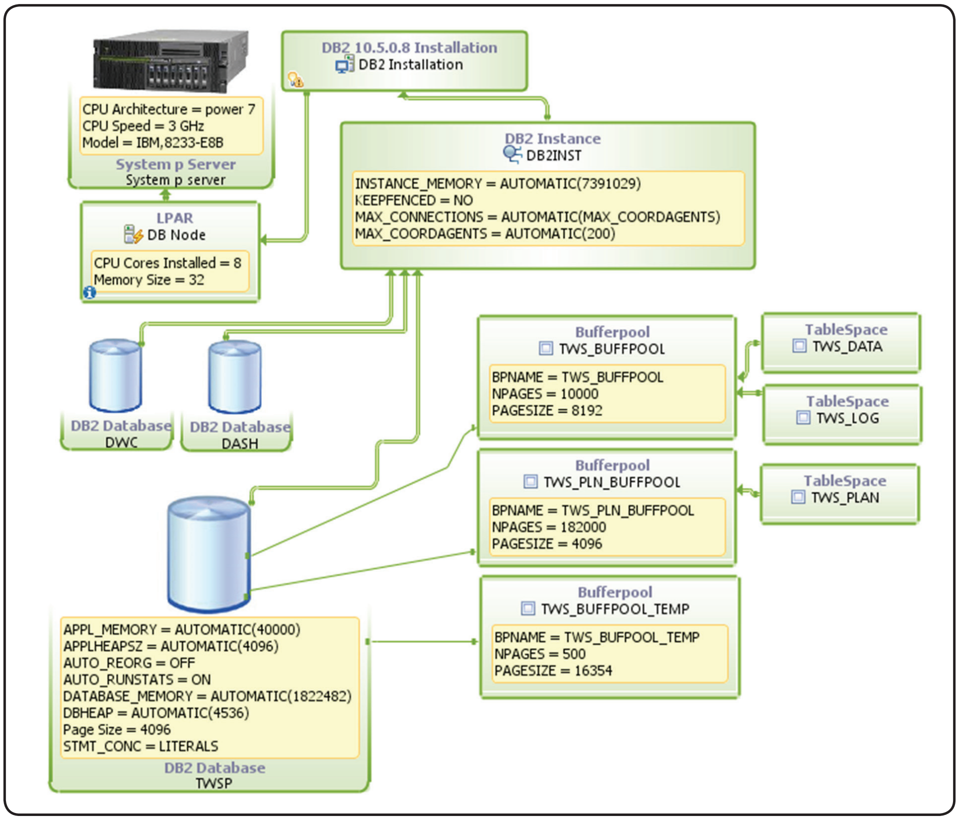

Figure 4. Database node configuration

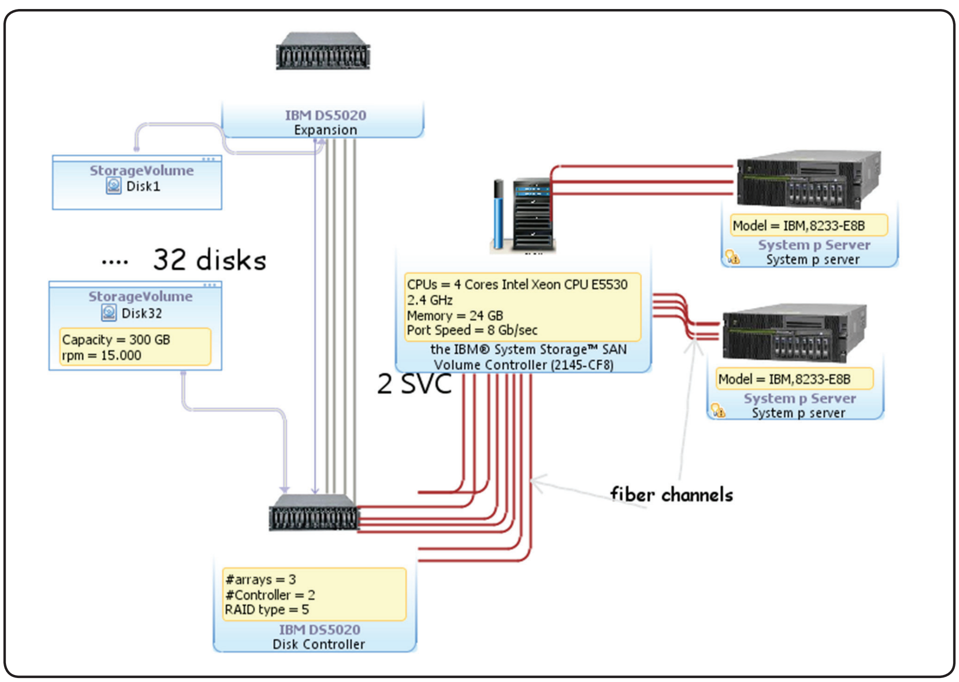

Figure 5. Storage solution

3.3 Test Tools

Rational Performance Tester (RPT) version 8.7.0.2 was used to generate traffic and run a multiple user scenario. RPT also provides a response time for each HTTP action on the browser by reporting the time spent on the server to process the request. RPT cannot determine how much time the browser spent processing the data to be interpreted.

Standard monitoring tools and methodologies were used, such as nmon and IBM Support Assistant 5.0 – Garbage Collection and Memory Visualizer. IOzone version 3.434 was used to benchmark storage throughput.

The Perfanalyst tool v. 1.1.4 was used to control the database configuration and to analyse the DB2 snapshot.

WebSphere Application Server Performance Tuning Toolkit v.2.0 is an intelligent toolkit which helps in tuning the performance of WebSphere Application Server.

All single user tests against the Dynamic Workload Console were performed in the node described in the following table:

<table>
<tr><th>Model</th><th>CPU</th><th>Browser</th></tr>
<tr><td rowspan="3">Lenovo W541</td><td rowspan="3">an Intel™ Core i7-4710MQ CPU @ 2.50 GHz ---- 8 logical cores</td><td>Mozilla Firefox ESR 45.6.0</td></tr>
<tr><td>Microsoft® Internet Explorer 11.0.38</td></tr>
<tr><td>Google Chrome 55.0</td></tr>
</table>

Table 3. DWC client configuration

3.4 Test Benchmarks and Results

3.4.1 Scheduling workload

This section reports the actual applied workload as specified in Table 1. Figures 6-11 represent each component of the workload in terms of outgoing throughput (black and yellow lines) and cumulative actual jobs schedule (blue solid area) with respect to (time) plan execution. The latter is merely the integral function of throughput.

The workload is homogenously distributed among fault-tolerant and dynamic agents and with respect to previous release improvements, no queuing in the schedule activity was detected. The indicator of this is the complete overlapping of actual scheduled jobs (blue solid graph) versus the planned one (red solid graph). Figure 11 shows the behavior of message logs triggered by event-driven workload automation for both workload scheduler objects and file creation.

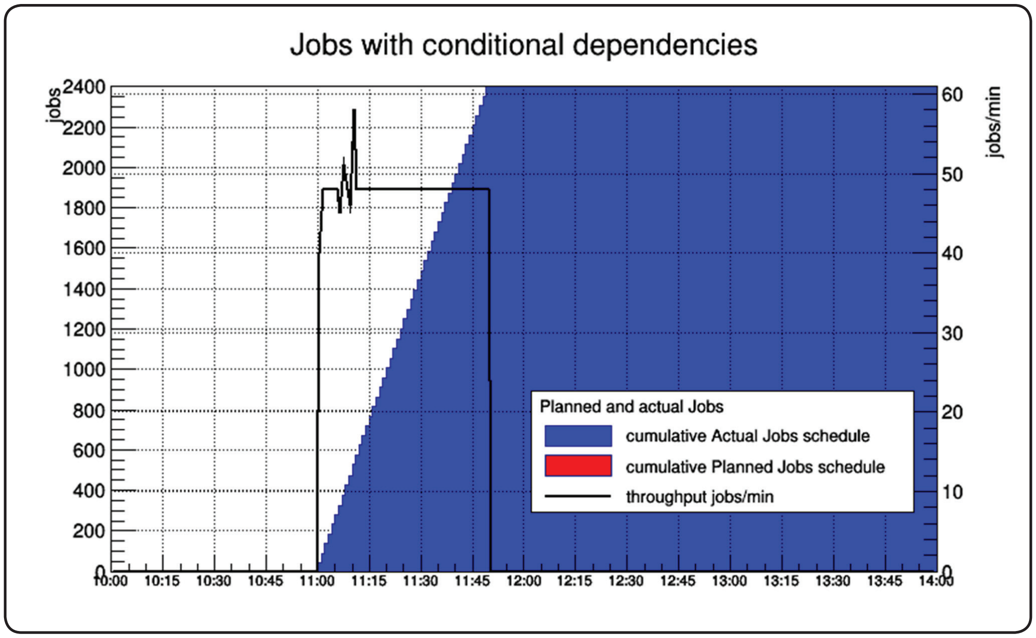

Figure 6. 3,200 jobs with conditional dependencies with 800 suppressed jobs

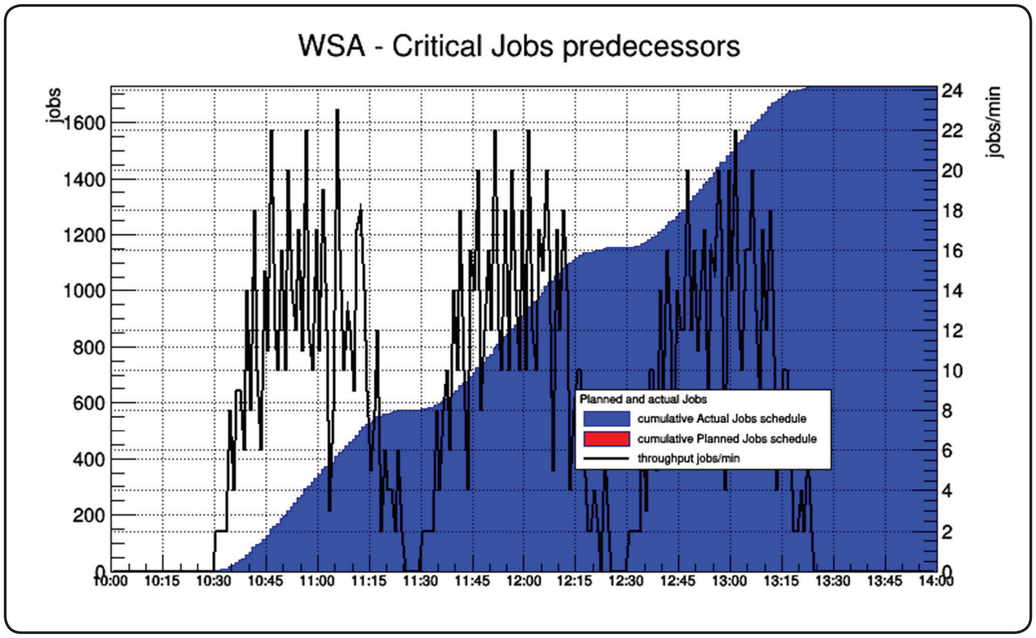

Figure 7. Critical jobs network (predecessors)

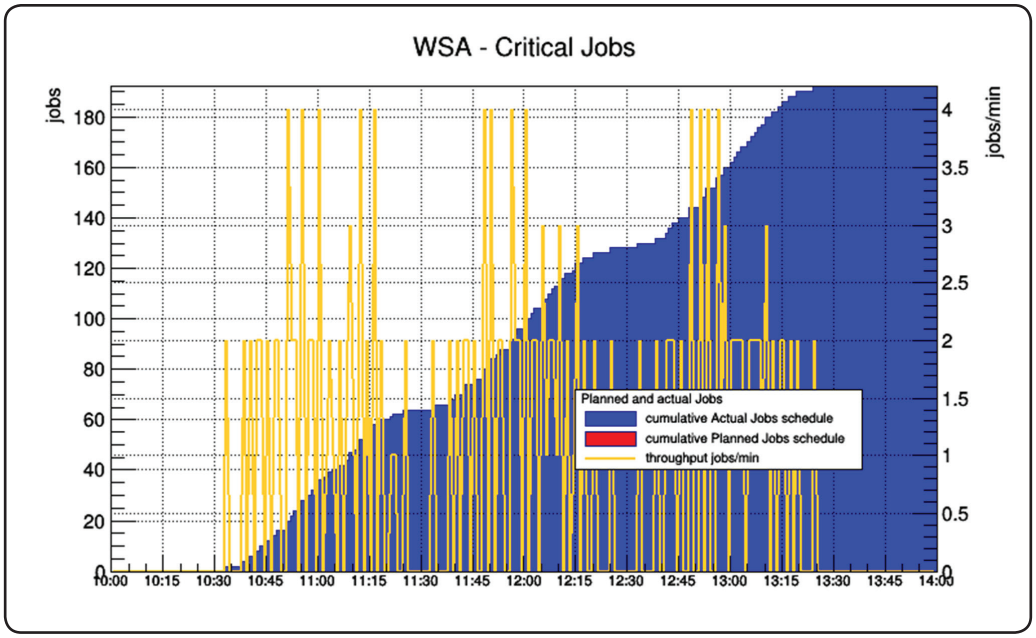

Figure 8. Critical jobs

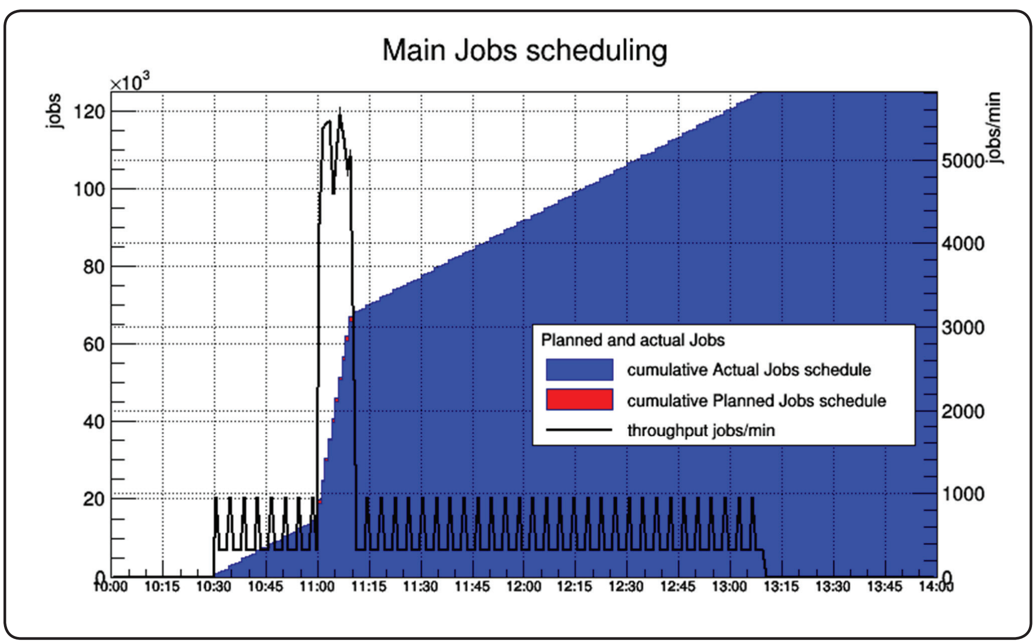

Figure 9. Main workload composed of a baseline of 480 jobs/min and a peak of 5,280 jobs/min

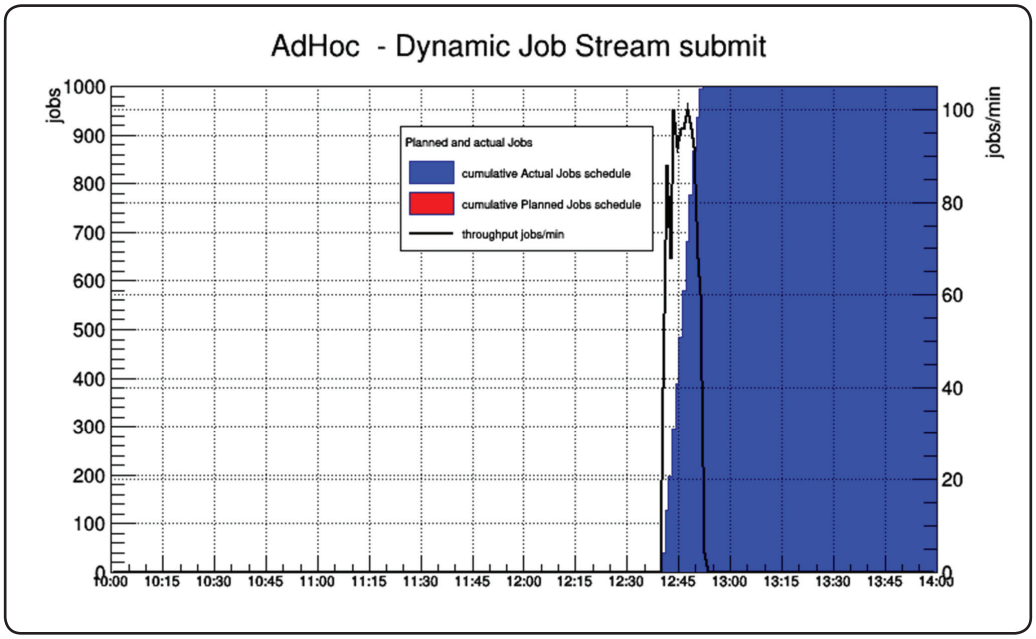

Figure 10. “conman sbs” job stream submission

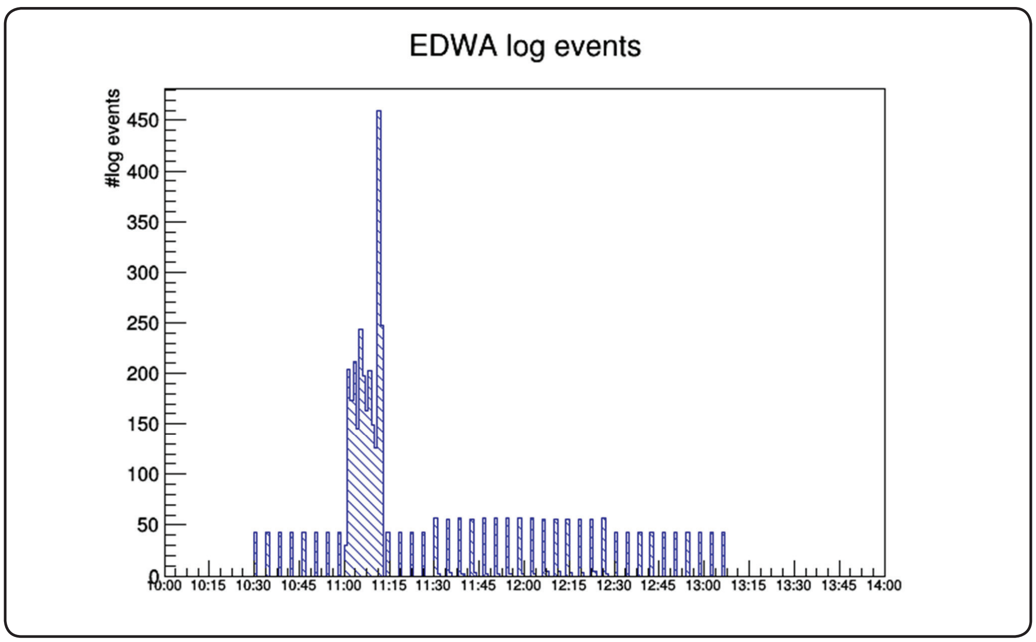

Figure 11. Event -driven triggered message log

The throughput analysis confirms the performance and scalability levels assured in the previous release.

3.4.2 Modeling graphical view

The Workload Designer graphical view was redesigned to enhance the user experience. The new design helps the user accomplish tasks easily and efficiently. Simple shapes to easily identify objects have been used, new icons to improve the interaction and quickly identify actions have been created, new colors and background to better visualize the objects have been applied.

The modeling graphical view for jobs and job streams was implemented within a new client base framework. Most of the previous master workload was moved to the client browser. That includes object relationship computation and graphical rendering. This important architectural change increases the concurrency for Workload Scheduler operators accessing the modeling graphical view.

Several job stream types were tested as reported in the following table:

Workload	Internal Jobs	Internal Job Dependencies
JS1	200	235
JS2	1000	950
JS3	1000	1900

Table 4. Job stream objects used for graphical view workload

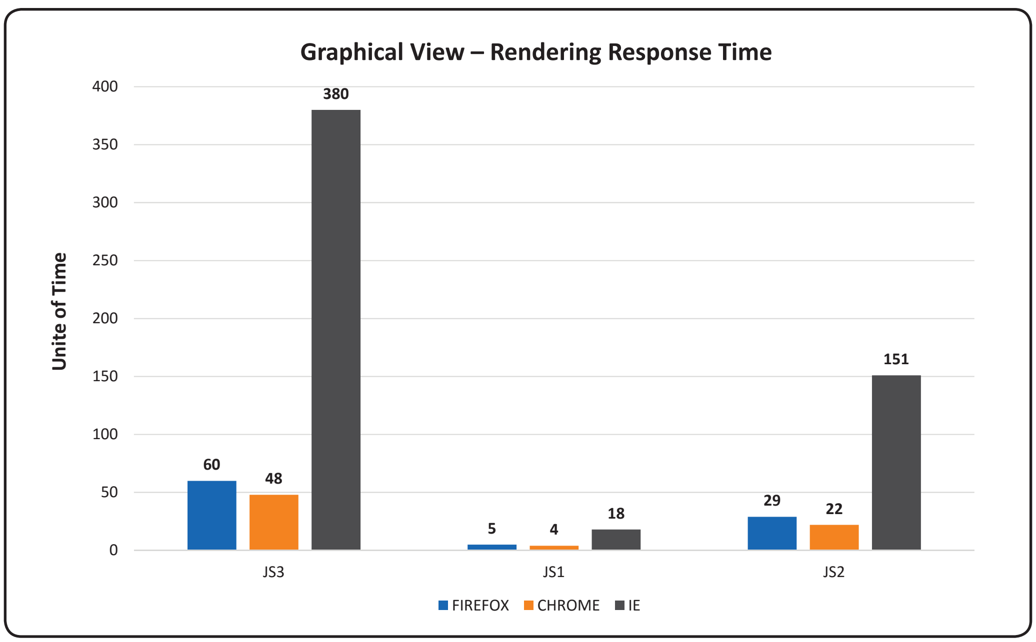

Figure 12. Rendering time for graphical view with different job streams and different browsers

Given a number of jobs included in a job stream, the rendering time appears to be almost linearly dependent on the total number of dependencies. This is due to the layout computation time. Tests were performed as a single user on the node described in Table 3.

Results show how performance is strictly related to the specific browser. Chrome performs better in this scenario.

Memory consumption is also not negligible. It was detected using the performance monitor high watermark for browser process.

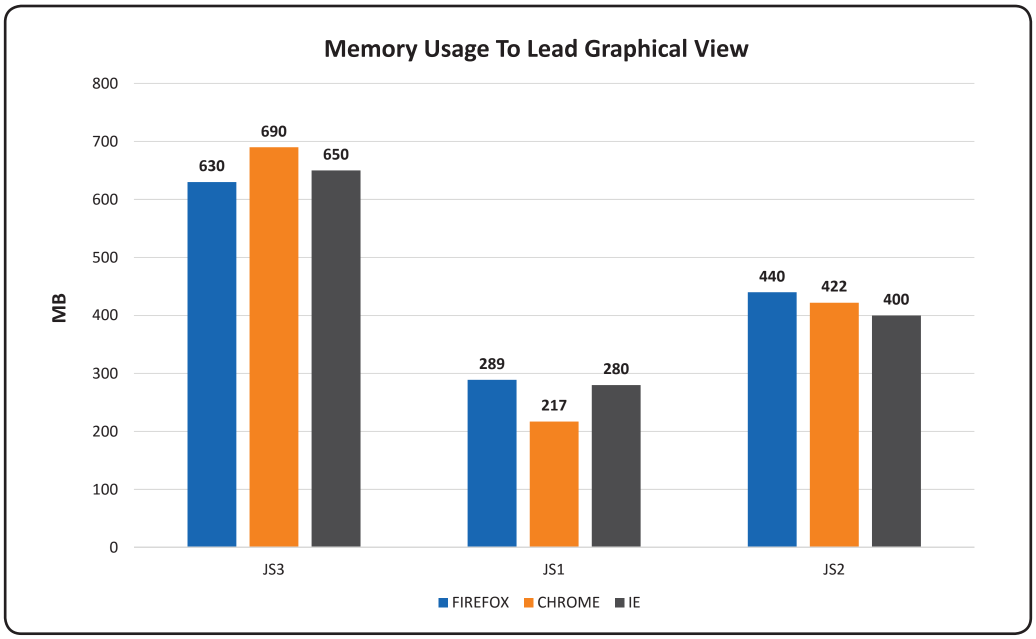

Figure 13. Memory usage to load graphical view by browser

The following screenshot gives a rough idea of the complexity of the layout of the job streams used in this benchmark.

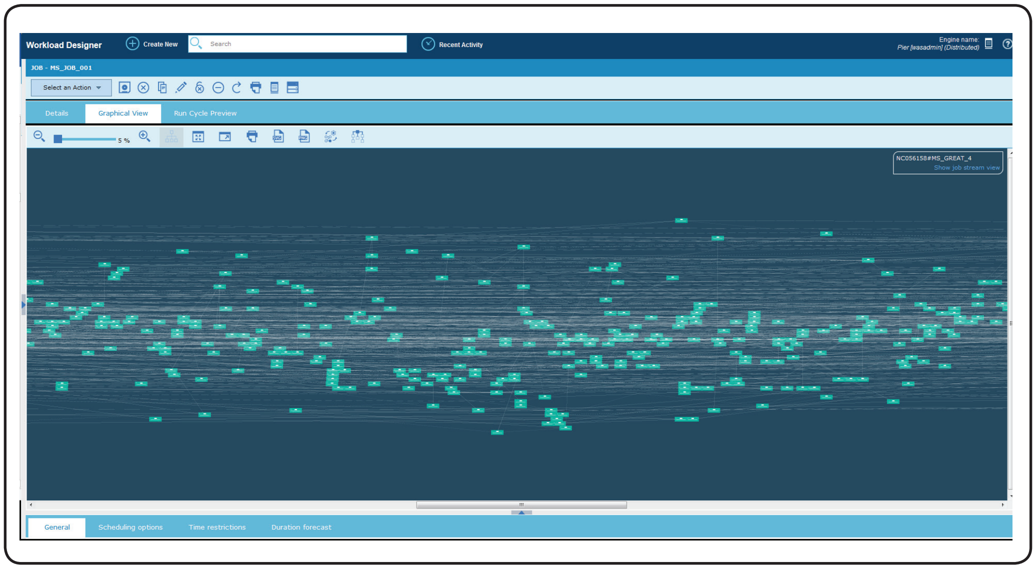

Figure 14. JS3 Workload Designer graphical view

It could be argued that the feasibility of handling such objects in a graphical framework, but these tests were designed in this way to push the new capability to the limit.

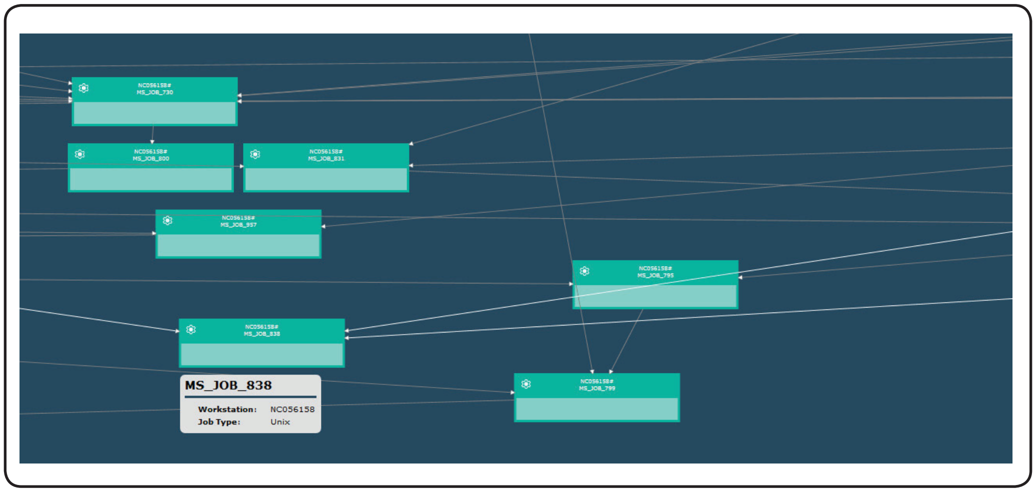

Figure 15. Details of some jobs belonging to the JS3 job stream

3.4.3 What-if analysis

The “What-if analysis” feature delivered in the version 9.3 product release, is a predictive capability that helps operators to forecast the impact of one or more jobs statuses on the plan execution. In the current release, this feature was enhanced to support the conditional dependencies feature.

In this context, an overall behavior of this feature was evaluated including a comparison for the impact of conditional dependencies. Two different sets of tests were applied to emphasize the rendering time behavior with relation to a number of increasing objects to load, conditional dependencies, object status in plan and the client browser.

<table>
<tr><th colspan="6">Workload Variation</th></tr>
<tr><td rowspan="2">Job stream chain</td><td rowspan="2">With/out 5% conditional dependencies</td><th colspan="2">Lenovo W541</th><th colspan="2">VM Client</th></tr>
<tr><td>Firefox</td><td>I.E.</td><td>Firefox</td><td>I.E.</td></tr>
<tr><td rowspan="2">Single job stream (JS2 of Table 4)</td><td>With/out 5% conditional dependencies</td><td>Firefox</td><td>I.E.</td><td></td><td></td></tr>
<tr><td>Running/Complete in Plan</td><td>Firefox</td><td>I.E.</td><td></td><td></td></tr>
</table>

Table 5. "What-if analysis" feature workload variation

The designed workload consists of a chain of 10 job streams (connected through dependencies). The higher is the number of the job stream selected in the chain, the greater is the number of objects to process, by selecting "first level predecessors" in the what-if panel using the right-click option.

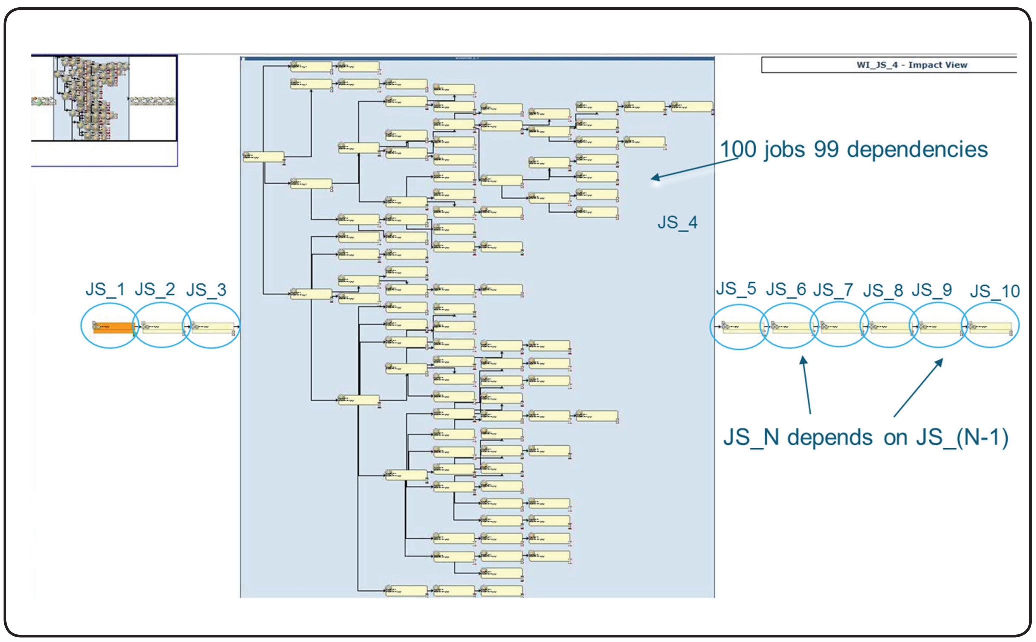

Figure 16. Job stream chain used in the What-if analysis performance test

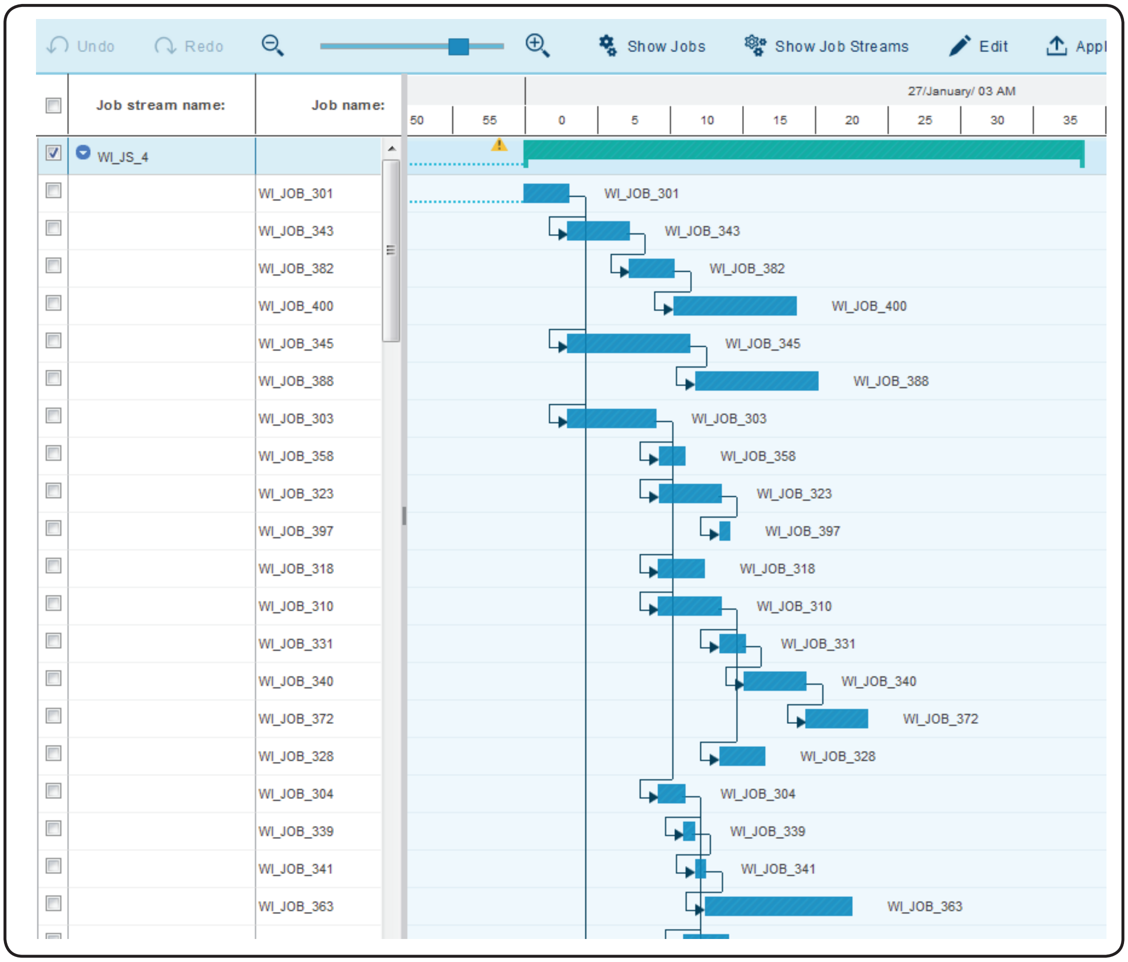

Figure 17. What-if analysis GANTT chart view of jobs

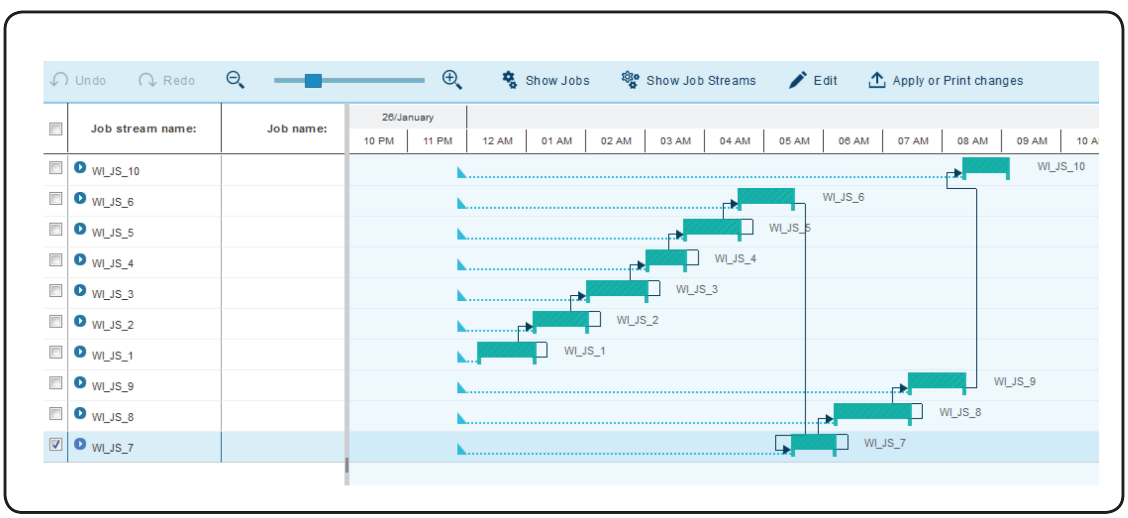

Figure 18. What-if analysis GANTT chart view of job streams collapsed

Test were performed both on Internet Explorer and Firefox as a single user in the environment described in Table 3.

In addition, a test on a VM node was performed to demonstrate how computational capabilities of the machine hosting the client could impact the browser rendering time.

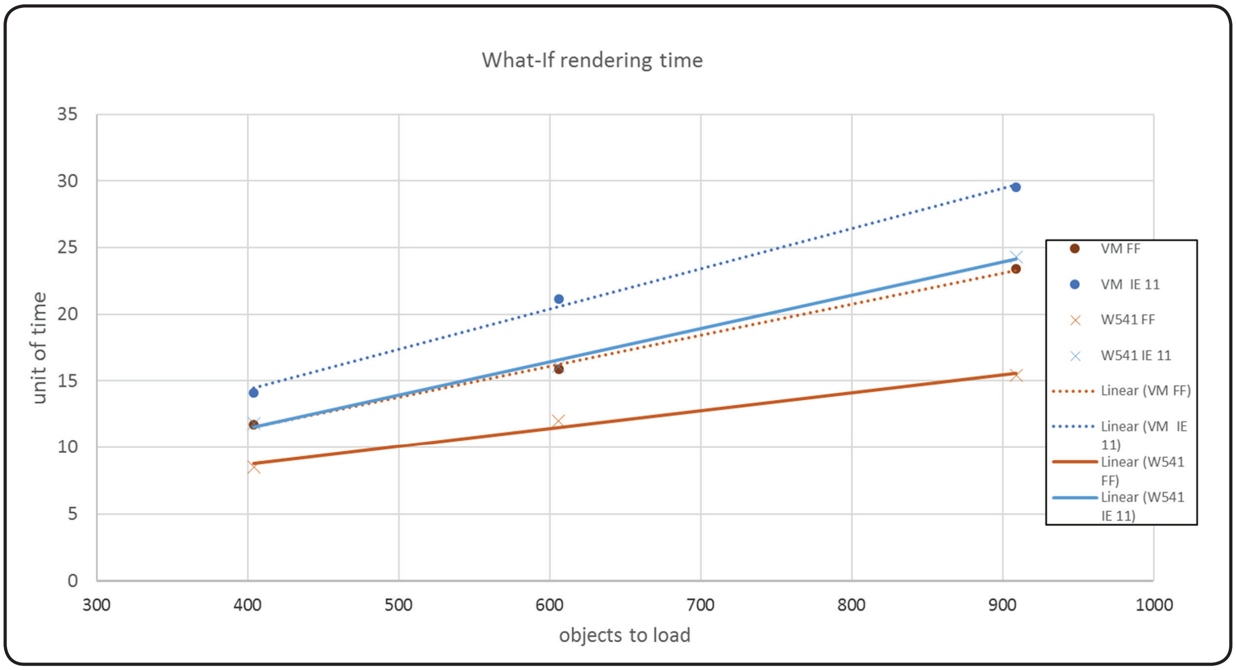

Figure 19. What-if analysis rendering time varying the number of objects to load, browser and client machine computational capacity

As can be seen in Figure 19, the response time is linearly dependent on the number of objects to load and, in addition, it is strictly dependent on the browser and computational capacity of the client node.

What follows are the results of the performance scenario related to different workloads including conditional dependencies and different workload scheduler job stream statuses in plan: running or complete.

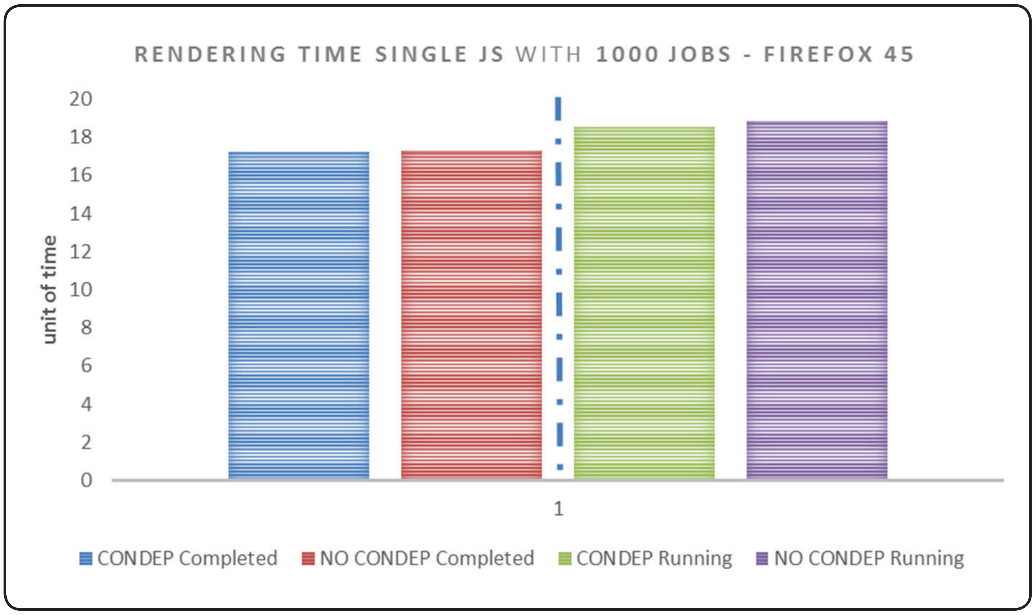

Figure 20. Rendering time in the Firefox browser comparing with/out conditional dependencies and different object statuses (completed/running)

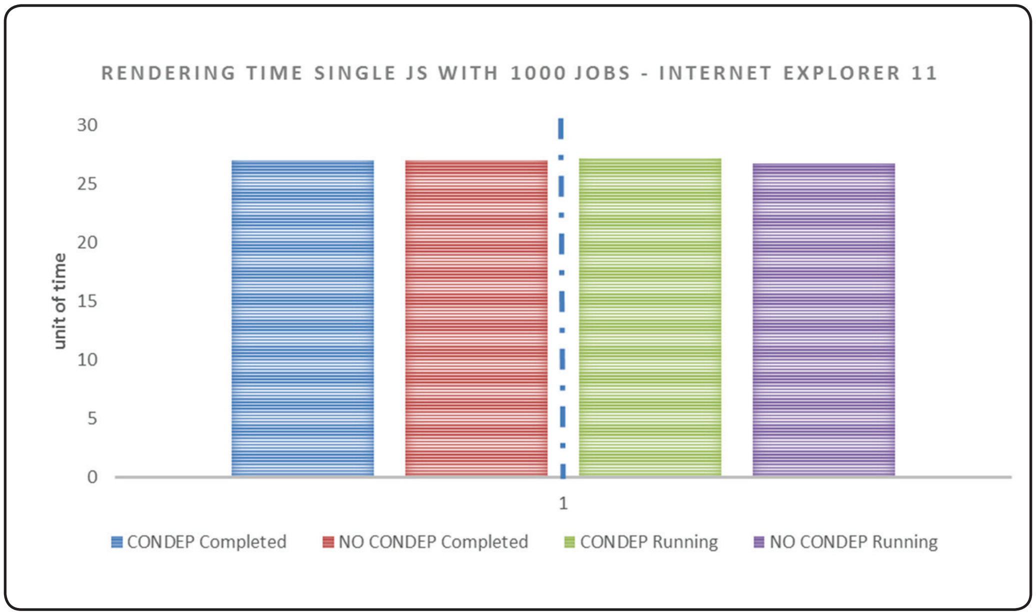

Figure 21. Rendering time in the Internet Explorer browser comparing with/out conditional dependencies and different object statuses (completed/running)

The main results of these tests can be summarized as follows:

- There are no differences between the case with or without conditional dependencies
- Firefox 45 performs better than Internet Explorer 11
- No significant differences were found for the rendering time between the Completed and Running object statuses

3.4.3.1 What-if analysis concurrency issue

Concurrent access to the What-if analysis view of several different archived plans with the WSA enabled revealed a performance and scalability issue. Each archived plan request causes the critical network to be loaded in memory impacting the engine application server both for CPU and memory consumption.

The test was done with 6 concurrent users performing their What-if analysis requests at the same time. The following diagram shows the Used Heap for the engine application server.

Figure 22 shows how the used heap reached the value of 3.15 GB when the What-if analysis concurrent requests were performed. This memory is used by the engine to build the critical network for the 6 different archived plans. This behavior could be critical for potential out of memory on the application server. The first 9.4.0.0 fix pack will provide a fix to improve the performance and to also mitigate strongly the risk of the engine JVM out-of-memory issue.

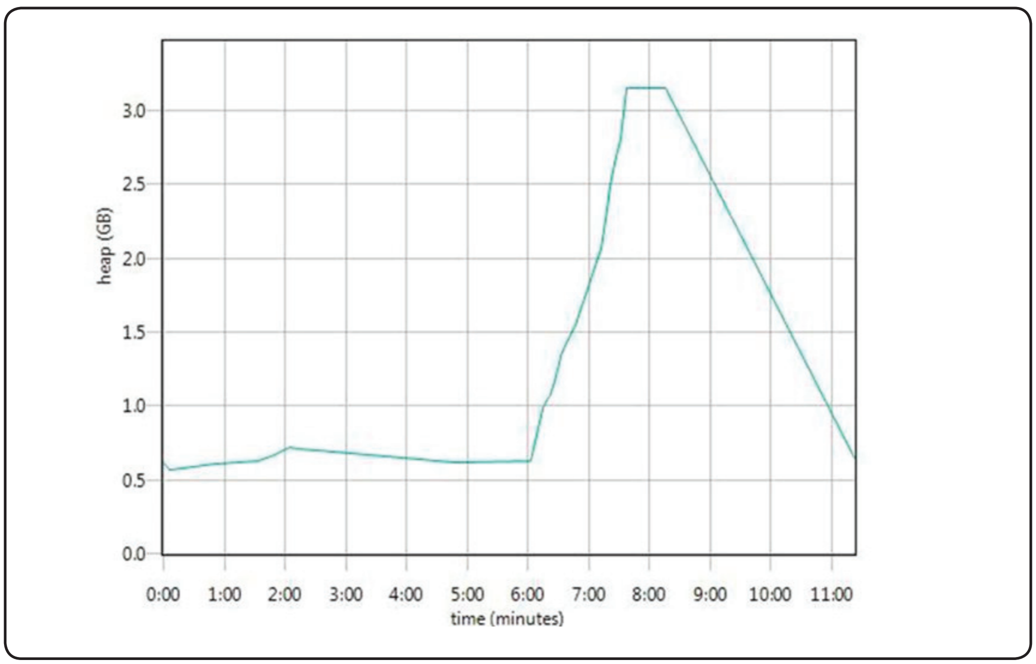

Figure 22. JVM used heap (after collection) during the creation of 6 different critical network on archived plans

3.4.4 Auditing feature

The new auditing feature provides versioning and rollback functions for all scheduling objects. Workload Scheduler administrators, operators, and schedulers can review all changes to scheduling objects, both in the database and in the plan, discover which user performed a specific change, and the time and date of when the change was performed. Administrators can also require that users provide a justification for the changes they make. Providing a justification consists of filling in the fields of a pop-up panel in the user interface.

Administrators can maintain an audit trail of each and every operation performed in the environment and generate a report.

The scope of the following performance benchmark is to verify if there was any significant performance degradation when the auditing of the information available in the database and in the plan is enabled.

To evaluate this feature, additional actions triggering auditing operations were added by means of Rational Performance Tester to execute 3 different test scenarios from the Dynamic Workload Console:

	Scope	Description	Auditing events
Scenario #1	Modeling (enDbAudit=1)	Create 2 new jobs and add them into a new job stream	227 new jobs and job streams
Scenario #2		Modify an existing job stream	57 unlocks 57 saves
Scenario #3	Plan (enPlanAudit=1)	Hold and release a job belonging to a specific job stream	211 hold events 211 release events

Table 6. Dynamic Workload Console scenarios evaluating the auditing feature

The workload in Table 6 is applied on top of the scheduling workload outlined in Section 3.4.1 which lasted 3 hours from 10:30 to 13:30. The test demonstrated that key performance indicators were impacted by less than 5%.

The results for both auditStore=FILE and auditStore=BOTH configurations outline an undetectable impact on database plan status update throughput (mirroring), the dynamic agent schedule throughput and resources utilization on Engine and Database Server.

A rough estimate of the impact of the auditing feature with regard to table space allocation is about 1 KB per auditing event (independent of whether or not the justification option is enabled). Some consideration must be made for the auditing historical data cleanup managed by the (“auditHistory / ah”) optman parameter which determines how many days audit records are maintained before they are deleted. The default value is 180 days.

3.4.5 User interface scenarios

Also in this context, the main objective of this test was to confirm the user experience improvements in terms of throughput and capacity and scalability properties. Differently from previous release tests, the workload was reworked:

Area	Scenario	Percentage	
Monitoring	Perform a monitor job query to search for a specific job to view job properties and predecessors.	35%	
	Perform a job stream query to search for a specific job stream and eventually to retrieve the job log or to show the Job Stream graphical view.	35%	With random 10% graphical activities and 10% retrieving job log
	Workload Dashboard initialization and navigation through some portlets (available workstation, late jobs, high risk jobs) present in the dashboard.	5%	
Mobile	Navigate through the Self-Service Dashboard for monitoring purposes.	10%	
	Navigate through the Self-Service Catalog to submit a service and to monitor its completion status.	10%	
Modeling	Navigate through the “Workload Designer” application to create 2 new job definitions and 1 job stream definition.	3%	With auditing enablement
	Navigate through the “Workload Designer” to search a job and to edit it and save	2%	

Table 7. Dynamic Workload Console test scenarios

For the user interface scenarios, a 4-node high availability configuration was used to support 700 concurrent users (175 users per node). Each user in the automation framework (Rational Performance Tester) logs in and completes three transactions before logging out and reentering again with different credentials. The delay between each transaction is controlled by the framework to have a frequency of:

20 transactions/hour per user

Test of 700 users generates an overall concurrency in the steady state of around:

38 pages/second

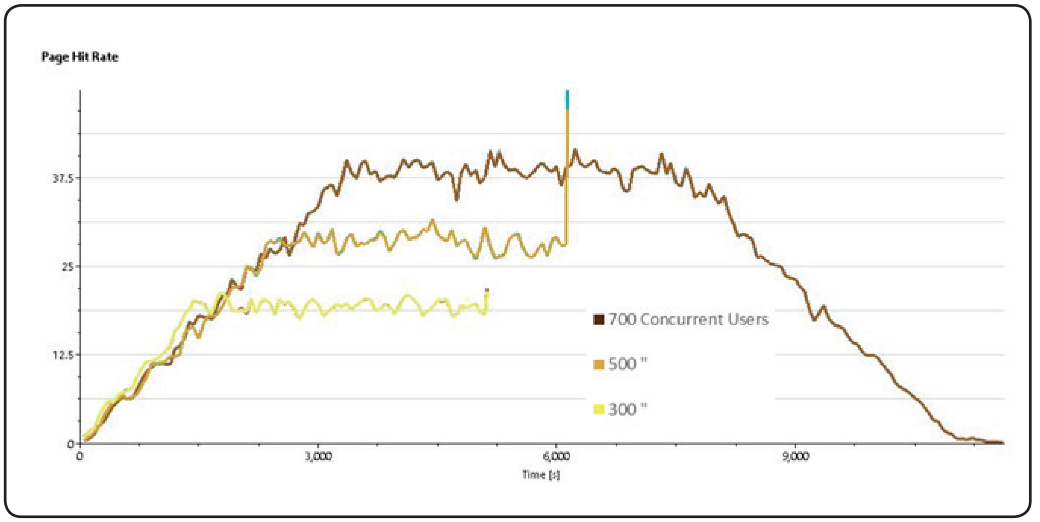

Figure 23. Page hit rate (page/second) caused by a 700-500-300 user test workload. For the scenario with 700 users, the logged-in users exited uniformly

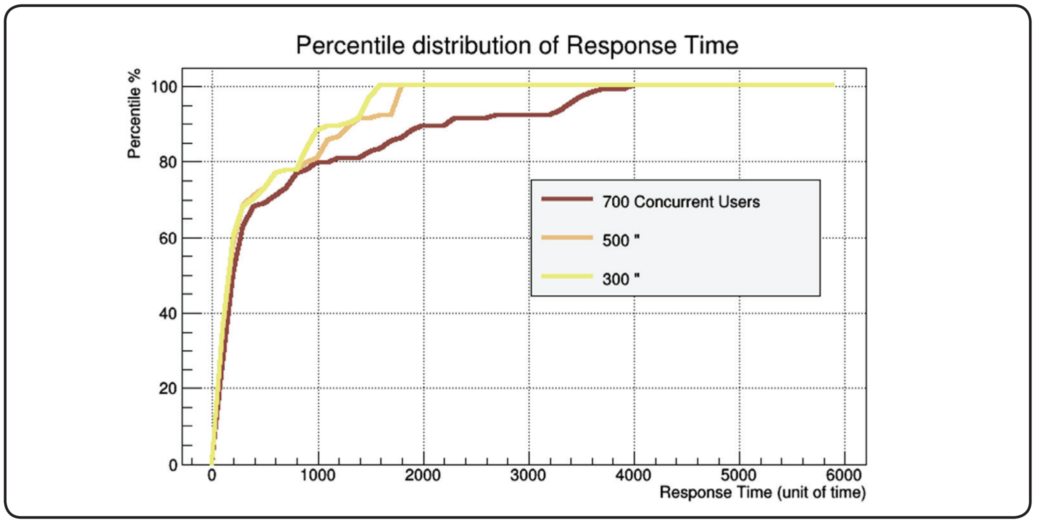

Figure 24. Percentile distribution of response time against 300, 500 and 700 concurrent users

The test run with 500 users was performed in a configuration with 3 Dynamic Workload Console nodes and a heap size of 4 GB. Instead, the test run with 700 concurrent users was performed in a configuration with 4 Dynamic Workload Console nodes and a heap size of 6 GB.

The following are some recommendations to maximize the performance and reliability of the product when there are 300 or more concurrent users:

1. Add a couple of database indexes on column NAME for the Dashboard Application Services Hug (DASH) database tables NODES and STORES to resolve a known issue that will be fixed in the next Jazz for Service Management release.
2. Ensure you do not preserve more than 1,000 records of service requests in the Self-Service Catalog application history.
3. Archive job reports on the agents regularly to avoid impact on performance when retrieving job log transactions from the server side.

3.4.5.1 Native memory issue

The Dynamic Workload Console concurrency test reveal unexpected behavior in the native memory usage trend. The class loader objects continue to be stored in the native memory until a global collection is triggered in the tenured heap area (gencon policy). Since the garbage collection policy (gencon) is tuned and optimized to reduce the frequency of global collection, the side effect of native memory consumption could lead to an out of memory issue of the Dynamic Workload Console JVM. To solve this issue, the following parameter was added to the Dynamic Workload Console JVM arguments:

Xgc:classUnloadingKickoffThreshold=1000

This parameter forces a global collection when the number of class loader reaches the specified threshold.

If the number of concurrent users per node is greater than 50, the suggestion is to set this JVM argument to the value 10,000.

4. RECOMMENDATIONS

4.1 CPU Capacity

All tests described in this document were executed on IBM Power7 8233-E8B 3GHz processors assigned exclusively to LPAR (no shared pools or capping feature were applied). While planning the correct CPU sizing, the information provided in Table 11 could be a reference point from which to start. The validity of the superposition property that allows us to assume that the resource usage can be considered as the sum of the UI (DWC) usage plus the core scheduling usage was demonstrated .

4.2 Storage

The scope of this document is not to suggest a specific storage solution, but rather the relevance of I/O capacity as outlined in the *"IBM Workload Scheduler Version 9.3.0.1 Performance and Capacity Planning Guide"* document in relation to product performance. Throughputs presented in Figure 25 could be used as a reference to maximize Workload Scheduler performance while planning a solution and the output of I/O industry standard benchmark, such as IOzone, as key performance indicators to compare with that reference.

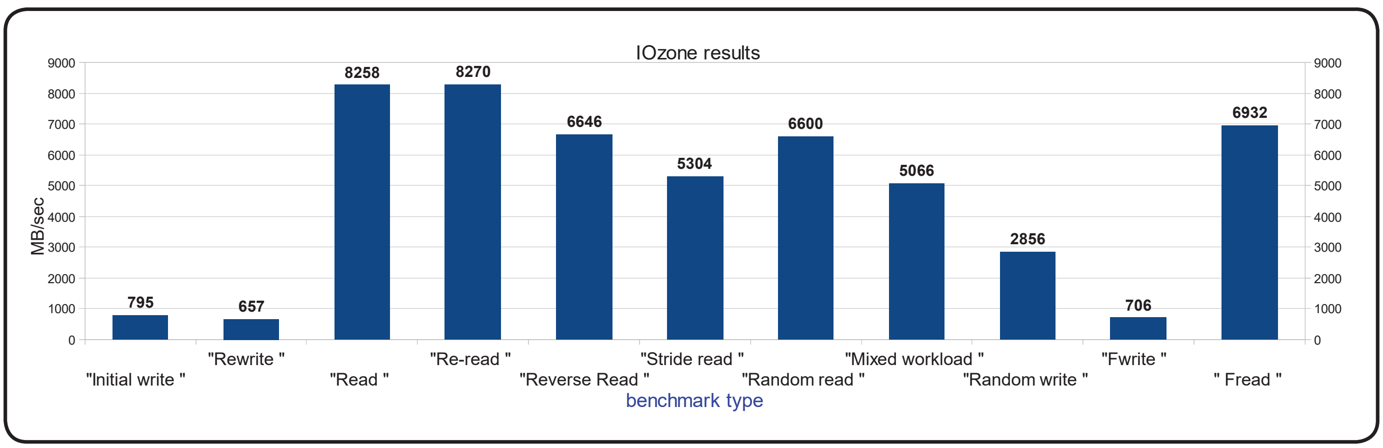

Figure 25. IOzone benchmark for storage solution

4.3 Memory

RAM size is strongly impacted by the JVM heap size settings whose suggested configuration can be found in the following tables:

Concurrent users range x DWC node	1 – 50	50 -100	100 -200
DWC heap size	1 GB	2 GB	4 - 6 GB

Table 8. Dynamic Workload Console WebSphere Application Server heap configuration

Schedule (jobs per min)	1 – 50	50 -100	100 -200	>200
WS Engine heap size	1 GB	1.5 GB	2 GB	4 GB

Table 9. Engine WebSphere Application Server heap configuration

In addition to the above memory requirements, the native memory for the Java™ process and Workload Scheduler process should be taken into consideration.

4.4 Tunings and Settings

The following parameters were tuned during the tests. These appliances are based on common performance best practices, also used in previous releases, and tuning activities during the test execution.

	PARAMETER		VALUE	COMMENT
Dynamic Workload Console Node	Dynamic Workload Console configuration settings repository (see https://www.ibm.com/support/knowledgecenter/SSGSPN_9.4.0/com.ibm.tivoli.itws.doc_9.4/distr/src_ad/awsaddwcanddb2.htm)		Use database as settings repository	It is strongly recommended to adopt this configuration to allow acceptable UI performance.
	WebSphere Application Server WC Thread Pool Size		300	Should be adjusted with number of concurrent users accordingly.
	WebSphere Application Server JVM max heap = min heap		Required: 4,096 for [100, 200] users per node Suggested: 6,144 for [150, 200] users per node	
	WebSphere Application Server JVM options		-Djava.awt.headless=true -Dsun.rmi.dgc.ackTimeout=10000 -Xdisableexplicitgc -Xgcpolicy:gencon -Xmn1536m -Xjit:exclude={org/mozilla/javascript/UintMap.rehashTable*} -Xgc:classUnloadingKickoffThreshold=10000	-Xmn parameter value should be ¼ of total heap size. This parameter should be set to 1,536m if heap = 6,144.
	WebSphere Application Server JDBC max Connections		300	
Workload Scheduler engine	WebSphere Application Server JDBC max Connections		300	
	WebSphere Application Server JVM max heap = min heap		2048 - 4096	
	WebSphere Application Server JVM options		-Djava.awt.headless=true -Dsun.rmi.dgc.ackTimeout=10000 -Xdisableexplicitgc -Xgcpolicy:gencon –Xmn 512m	- Xmn 1,024m if heap size = 4,096
	localopts	batchman settings	bm check deadline = 0 bm check file = 120 bm check status = 300 bm check untils = 300 bm late every = 0 bm look = 10 bm read = 10 bm stats = off bm verbose = off	
DB	LOGPRIMARY		200	780 MB total transaction log space
	LOGFILSIZ		1000	
	KEEPFENCED		NO	
	dbMAX_CONNECTION		AUTOMATIC	
	STMT_CONC		LITERALS	This setting optimizes query executions and reduces CPU usage.
	Db APPL_MEMORY, APPLHEAPSZ, DATABASE_MEMORY, DBHEAP		AUTOMATIC	
	Db AUTO_RUNSTAT		ON	
	AUTO_REORG		OFF	
	TWS_PLN_BUFFPOL	NPAGES	182000	
		PAGESIZE	4096	
	TWS_BUFFPOOL_TEMP	NPAGES	500	
		PAGESIZE	16384	
	TWS_BUFFPOOL	NPAGES	10000	
		PAGESIZE	8192	
Dynamical Workload Broker	JobDispatcherConfig.properties	Historical data management	MoveHistoryDataFrequencyInMins=720	
		Queue settings	Queue.actions.0 = cancel, cancelAllocation, cancelOrphanAllocation Queue.size.0 = 10 Queue.actions.1 = reallocateAllocation Queue.size.1 = 10 Queue.actions.2 = updateFailed Queue.size.2 = 10 Queue.actions.3 = completed Queue.size.3 = 30 Queue.actions.4 = execute Queue.size.4 = 30 Queue.actions.5 = submitted Queue.size.5 = 30 Queue.actions.6 = notification Queue.size.6 = 30	
	ResourceAdvisorConfig.properties	MaxAllocsPerTimeSlot	1000	
		TimeSlotLength	10	
		MaxAllocsInCache	50000	

Table 10. Main configurations and tunings

5. CAPACITY PLAN EXAMPLES

In the context of this document, the number of key parameters used to identify the workload was kept as simple as possible:

1. Number of concurrent users assuming a mixed scenario similar to the one described in 3.4.1;

2. Number of jobs to be scheduled;

3. Percentage of dynamic jobs to schedule.

With the above inputs, it is possible to forecast the resources needed to host the version 9.4.0.0 product. Internal fit functions were used to model the workload and resource usage relationship. A 65% CPU usage was the threshold considered before requesting additional core.

In this section, some examples of capacity planning are reported. Remember that all of the requirements are related to the PowerPC P7 platform; nevertheless, this information could be used as a reference point for different platform architectures.

	NODE	Core Capacity	Disk Throughput Read-Write (MB/sec)	Network Throughput Read-Write (MB/sec)	RAM Capacity (GB)
250K jobs (50% FTA +50% DYN) per day (175 jobs/min) 100 concurrent users					
3Nodes	WS-Engine	2	0-0.5	1-1	3
	RDBMS	1	2-0.5	0.5-1.5	5
	DWC	2	0-0.1	1.2-1	6
500K jobs (50% FTA +50% DYN) per day (350 jobs/min) 100 concurrent users					
3Nodes	WS-Engine	2	0-1	0.9-2	4
	RDBMS	2	2.3-0.9	0.5-1.5	5
	DWC	2	0-0.1	1.2-1	6
750K jobs (50% FTA +50% DYN) per day (485 jobs/min) 100 concurrent users					
3Nodes	WS-Engine	3	0-1.3	1.6-1.3	4
	RDBMS	3	2.3-1.2	1-2.2	5
	DWC	2	0-0.1	1.2-1	6
10K jobs (50% FTA +50% DYN) per day (8 jobs/min) 20 concurrent users					
1Node	WS-Engine RDBMS DWC	1	0.5-0.1	0.5-0.7	5

Table 11. Capacity planning samples

The above capacity planning examples refer to the workload described in section Table 1. In particular, they are based on job scheduling performed on a number of workstations where 50% of those are dynamic agent workstations. If the ratio changes, the engine CPU capacity requirement changes. For example, assuming that all agents are dynamic (100%) the following configuration should be considered:

	NODE	Core Capacity	Disk Throughput Read-Write (MB/sec)	Network Throughput Read-Write (MB/sec)	RAM Capacity (GB)
500K jobs (100% DYN) per day (350 jobs/min) 100 concurrent users					
3Nodes	WS-Engine	3	0-1	0.9-2	4
	RDBMS	3	2.3-0.9	0.5-1.5	5
	DWC	2	0-0.1	1.2-1	6

Table 12. Impact on workload with 100% dynamic agent job scheduling

It must be made clear that an elementary estimation of disk throughput (MB/sec) is not sufficient to design an appropriate storage solution. It is recommended to take into consideration the benchmarks reported in section 4.2.

6. NOTICES

This information was developed for products and services offered in the U.S.A.

HCL may not offer the products, services, or features discussed in this document in other countries. Consult your local HCL representative for information on the products and services currently available in your area. Any reference to an HCL product, program, or service is not intended to state or imply that only that HCL product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any HCL intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-HCL product, program, or service.

HCL may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to HCL TECHNOLOGIES LIMITED email: products-info@hcl.com

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: HCL TECHNOLOGIES LIMITED PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR

FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. HCL may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-HCL Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this HCL product and use of those Web sites is at your own risk.

HCL may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact HCL TECHNOLOGIES LIMITED email: *products-info@hcl.com*

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by HCL under terms of the HCL License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-HCL products was obtained from the suppliers of those products, their published announcements or other publicly available sources. HCL has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-HCL products. Questions on the capabilities of non-HCL products should be addressed to the suppliers of those products.

All statements regarding HCL's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All HCL prices shown are HCL's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

7. TRADEMARKS

HCL, and other HCL graphics, logos, and service names including "hcltech.com" are trademarks of HCL. Except as specifically permitted herein, these Trademarks may not be used without the prior written permission from HCL. All other trademarks not owned by HCL that appear on this website are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by HCL.

IBM is a trademark or registered trademark of International Business Machines Corporation in the United States, other countries, or both. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

About HCL Technologies

HCL Technologies (HCL) is a leading global IT services company that helps global enterprises re-imagine and transform their businesses through Digital technology transformation. HCL operates out of 32 countries and has consolidated revenues of US$ 7 billion, for 12 months ended 31st December, 2016. HCL focuses on providing an integrated portfolio of services underlined by its Mode 1–2–3 growth strategy. Mode 1 encompasses the core services in the areas of Applications, Infrastructure, BPO and Engineering & R&D services, leveraging DRYiCETM Autonomics to transform clients' business and IT landscape, making them 'lean' and 'agile'. Mode 2 focuses on experience-centric and outcome-oriented services such as Digital and Analytics Services (BEYONDigitalTM), IoT WorKSTM, Cloud and Security, utilizing DRYiCETM Orchestration to drive business outcomes and enable enterprise digitalization. Mode 3 strategy is ecosystem-driven, creating innovative IP-partnerships to build products and platforms business.

HCL leverages its global network of integrated co-innovation labs, and global delivery capabilities to provide holistic multi-service delivery in key industry verticals including Financial Services, Manufacturing, Telecommunications, Media, Publishing, Entertainment, Retail CPG, Life Sciences Healthcare, Oil & Gas, Energy & Utilities, Travel, Transportation & Logistics and Government. With 115,000 professionals from diverse nationalities, HCL focuses on creating real value for customers by taking 'Relationships Beyond the Contract'. For more information, please visit **www.hcltech.com**

About HCL Enterprise

HCL is a $7 billion leading global technology and IT enterprise comprising two companies listed in India – HCL Technologies and HCL Infosystems. Founded in 1976, HCL is one of India's original IT garage start-ups. A pioneer of modern computing, HCL is a global transformational enterprise today. Its range of offerings includes product engineering, custom & package applications, BPO, IT infrastructure services, IT hardware, systems integration, and distribution of information and communications technology (ICT) products across a wide range of focused industry verticals. The HCL team consists of over 115,000 ideapreneurs of diverse nationalities, who operate from 32 countries including over 505 points of presence in India. HCL has partnerships with several leading global 1000 firms, including leading IT and technology firms. For more information, please visit **www.hcl.com**