Standard Performance Evaluation Corporation (SPEC)

SPECjbb2015 Benchmark Run and Reporting Rules 1.02

7001 Heritage Village Plaza, Suite 225
Gainesville, VA 20155,
USA

Table of Contents

1. Introduction

This document specifies how the SPECjbb2015 benchmark is to be run for measuring and publicly reporting performance results. These rules abide by the norms laid down by SPEC in order to ensure that results generated with this benchmark are meaningful, comparable to other generated results, and repeatable, with documentation covering factors pertinent to reproducing the results. Per the SPEC license agreement, all results publicly disclosed must adhere to these Run and Reporting Rules.

To check for possible updates to this document, please see http://www.spec.org/jbb2015/docs/runrules.pdf.

1.1. Definitions

- **Machine**: Hardware (H/W) required to run an OS instance
- **Module**: H/W required to run an OS instance within a shared infrastructure
- **Node**: A single machine or a module
- **System**: A single stand-alone machine or one or more nodes sharing a common shared infrastructure such as a backplane, power-supplies, fans or other elements and are able to run one or more OS images. As example, a blade server with *n* nodes sharing a common shared infrastructure is a "Single system with *n* nodes". However if a node can run stand stand-alone it is considered a distinct system.
- **Single Host**: Single system running a single OS image
- **Multiple Hosts**: One or more systems running more than one OS image
- **System(s) Under Test (SUT)**: Those system(s) which form the basis of benchmark comparison

1.2. Philosophy

SPEC believes the user community will benefit from an objective series of benchmark results, which can serve as a common reference and be considered as part of an evaluation process. SPEC expects that any public use of results from this benchmark suite must be for the Systems Under Test (SUTs) and configurations that are appropriate for public consumption and comparison. For results to be publishable, SPEC requires:

- Proper use of the SPEC benchmark tools as provided.
- Availability of an appropriate full disclosure report (FDR).
- Availability of the Hardware and Software used (see section 2.6).
- Support for all of the appropriate protocols.

1.2.1. Applicability

SPEC intends that this benchmark measure the performance of systems providing environments for running server-side Java applications. It is not a Java EE benchmark and therefore it does not measure Enterprise Java Beans (EJBs), servlets, Java Server Pages (JSPs), etc.

While this benchmark was designed to be a measure of computer servers, SPEC acknowledges that it may also be possible to measure other classes of computing devices. Given the speed of technology advances in the industry, SPEC does not arbitrarily restrict the type of system on which the benchmark is measured. However, since it would be misleading to draw comparisons between systems that are intended for substantially different uses, this document includes rules to promote fair comparisons between systems that are intended for similar purposes.

Note that while it may be possible to run this benchmark on personal systems, SPEC provides a substantial suite of benchmarks intended for evaluation of workstations that should be considered (http://www.spec.org/benchmarks.html#gwpg).

1.2.2. Optimizations

SPEC is aware of the importance of optimizations in producing the best system performance. SPEC is also aware that it is sometimes difficult to draw an exact line between legitimate optimizations that happen to benefit SPEC benchmarks and optimizations that specifically target a SPEC benchmark. However, with the rules below, SPEC wants

to increase the awareness of implementers and end users of issues of unwanted benchmark-specific optimizations that would be incompatible with SPEC's goal of fair benchmarking.

- Hardware and software used to run the SPECjbb2015 benchmark must provide a suitable environment for running typical server-side Java applications. (Note: this may be different from a typical environment for client Java application)
- Software optimizations must generate correct code for a class of programs, where the class of programs must be larger than a single SPEC benchmark.
- Hardware and/or software optimizations must improve performance for a class of applications, where the class of applications must be larger than a single SPEC benchmark.
- The vendor encourages the implementation for general use.
- The implementation is generally available, documented, and supported by the providing vendor(s).

Furthermore, SPEC expects that any public use of results from this benchmark must be for configurations that are appropriate for public consumption and comparison. In the case where it appears that the above guidelines have not been followed, SPEC may investigate such a claim and take action in accordance with current policies.

1.3. Caveats

SPEC reserves the right to investigate any case where it appears that these guidelines and the associated benchmark run and reporting rules have not been followed for a public SPEC benchmark claim. SPEC may request that the claim be withdrawn from the public forum in which it appears and that the benchmark tester correct any deficiency in product or process before submitting or publishing future results.

SPEC reserves the right to adapt the benchmark codes, workloads, and rules of the SPECjbb2015 benchmark as deemed necessary to preserve the goal of fair benchmarking. SPEC will notify members and licensees whenever it makes changes to the benchmark and may rename the metrics. In the event that the workload and/or metrics are changed, SPEC reserves the right to republish, in summary form, "adapted" results for previously published systems, converted to the new metric. In the case of other changes, a republication may necessitate retesting and may require support from the original test sponsor.

Relevant standards are cited in these run rules as URL references, and are current as of the date of publication. Changes or updates to these referenced documents or URLs may necessitate repairs to the links and/or amendment of the run rules. SPEC will notify members and licensees whenever it makes changes to the suite.

1.4. Research and Academic Usage

Please consult the SPEC Fair Use Rule for Research and Academic Usage (http://www.spec.org/fairuse.html#Academic) for the SPECjbb2015 benchmark.

2. Run Rules

2.1. Measurement

The provided SPECjbb2015 benchmark tools must be used to run and produce measured the SPECjbb2015 benchmark results. The SPECjbb2015 benchmark metrics are a function of the SPECjbb2015 benchmark workload, and the defined benchmark control parameters. The SPECjbb2015 benchmark results are not comparable to performance metrics from SPECjbb2005, SPECjbb2013 or any other application.

2.2. Initializing and Running Benchmark

For guidance, please consult the latest User Guide on the SPEC's website (http://www.spec.org/jbb2015/docs/userguide.pdf).

2.3. Workload and Workload Components

The SPECjbb2015 benchmark exercises a Java application workload. A detailed description of the workload and its coponents can be found in the latest version of the design document (http://www.spec.org/jbb2015/docs/designdocument.pdf).

2.3.1. Benchmark Binaries and Recompilation

The benchmark kit includes tools for running the benchmark and reporting its results. The workload is written in Java; precompiled class files are included with the kit, so no build step is necessary. This software implements various checks for conformance with the run and reporting rules, therefore the SPEC software must be used.

The SPECjbb2015 benchmark binaries are provided in jar files containing the Java classes. Valid runs must use the provided jar files and these files must not be updated or modified in any way. While the source code of the benchmark is provided for reference, the benchmark tester must not recompile any of the provided .java files. Any runs that used recompiled class files are not valid and cannot be reported or published.

2.3.2. Manual Intervention

No manual intervention or optimizations to the controller, transaction injector(s), SUT(s), or their internal and external environments are allowed during the benchmark run.

2.3.3. Execution

The phases are described in detail in the latest Design Document on SPEC's website (http://www.spec.org/jbb2015/docs/designdocument.pdf).

2.4. Benchmark Control Parameters

There are many properties that control the operation of the SPECjbb2015 benchmark. The ones that are user-settable are listed below. For a compliant run, these properties may be set to values other than the default if they follow the allowed ranges in Table 2.4-1 and 2.4-2, and all other properties must not be changed from the specified values.

The SPECjbb2015 benchmark agents Controller (Ctr), Transaction Injector (TxI) and Backend can use different JVM parameters but in a multi-JVM environment, all JVM instances of Backend(s) must be identically configured and all TxI(s) must use the same JVM parameters unless due to technical reasons a configuration cannot meet these requirements.

Parameter name	Default Value	User setable(yes/no)
specjbb.group.count	1	Yes but must be >=1
specjbb.txi.pergroup.count	1	Yes but must be >=1
specjbb.comm.connect.client.pool.size	256	Yes
specjbb.comm.connect.worker.pool.min	1	Yes
specjbb.comm.connect.worker.pool.max	256	Yes
specjbb.comm.connect.selector.runner.count	0 (special value 0 means default setting)	Yes

specjbb.customerDriver.threads	64	Yes but must be >=64
specjbb.forkjoin.workers	2 x Runtime.getRuntime().availableProcessors()	Yes
specjbb.controller.rtcurve.warmup.step	0.10 (10%)	Yes but suggested <0.9

Table 2.4-1 Benchmark Properties that can affect performance

Parameter name	Default Value	User setable(yes/no)
specjbb.time.server	False	Yes for virtulized SUTs (Composite / Multi-JVM)
specjbb.controller.host	Localhost	Yes
specjbb.controller.port	24000	Yes
specjbb.controller.handshake.period	5000 (5 sec)	yes
specjbb.controller.handshake.timeout	600000 (600 sec)	yes
specjbb.comm.connect.timeouts.connect	60000 (60 sec)	Yes
specjbb.comm.connect.timeouts.read	60000 (60 sec)	Yes
specjbb.comm.connect.timeouts.write	60000 (60 sec)	Yes
specjbb.run.datafile	SPECjbb2015.data.gz	Yes
specjbb.heartbeat.period	10000 (10 sec)	Yes
specjbb.heartbeat.threshold	100000 (100 sec)	Yes
specjbb.controller.maxir.maxFailedPoints	3	Yes, but suggested ~3
specjbb.run.datafile.dir	. (current dir)	Yes
specjbb.mapreducer.pool.size	max(Runtime.getRuntime().availableProcessors(), specjbb.group.count * (specjbb.txi.pergroup.count +1))	Yes >=2

Table 2.4-2 Benchmark Properties that often don't affect performance

In addition to properties, the SPECjbb2015 benchmark has its own command line parameters. A complete list of options can be printed using option '-h'. Only the flags listed below in Table 2.4-3 are allowed for a compliant result.

Command line option	Option Input(s)	Remark
-m <mode>	[composite/ multicontroller/ distcontroller] for Controller, [txinjector] for TxI, [backend] for Backend	Launches a benchmark component
-G <alphanumeric ID>	<alphanumeric ID>	Group ID
-J <alphanumeric ID>	<alphanumeric ID>	JVM ID
-p <file>	Default: ./config/specjbb2015.props	Property file
-raw <file>	Defualt: ./config/template-[C/M/D].raw [C / M / D] picked based on run category	HW/SW configuration description C: Composite, M:MultiJVM, D:Distributed
-t <DIR>	<directory> (default: result)	Result directory
-l <num>	Default: 0 , <0, 1, 2, 3>	Report details (0: minimum, 3: most)
-skipReport	None (if used, user need to run reporter separately)	Skip report generation at the end of run
-s <file>	<binary log file of a run>	To re-generate submission raw file
-v	None (if used, too much output)	More verbose information

Table 2.4-3 Benchmark command line options

For any further explanation or help, user should refer to the SPECjbb2015 benchmark User Guide (http://www.spec.org/jbb2015/docs/userguide.pdf).

2.5. Optimization Flags

Both JVMs and native compilers are capable of modifying their behavior based on flags. Flags that do not break conformance to section 2.9 are allowed. All command-line flags used must be reported. All flags used must be documented and supported within the time frame specified in this document for general availability. At the time a result is submitted to SPEC, descriptions of all flags used but not currently publicly documented must be available to SPEC for the review process. When the result is published, all flags used must be publicly documented, either in the vendor's public documentation, in the disclosure, or in a separate flags file.

2.6. General Availability

The entire test-bed must be comprised of components that are generally available on or before date of publication, or must be generally available within three months of the first publication of these results.

Products are considered generally available if they are orderable by ordinary customers, and are shipped within a reasonable time frame. This time frame is a function of the product size, classification, and common practice. Some limited quantity of the product must have shipped on or before the close of the stated availability window. Shipped products do not have to match the tested configuration in terms of CPU count, memory size, and disk count or size, but the tested configuration must be available to ordinary customers. The availability of support and documentation of the products must be coincident with the release of the products.

Hardware products that are still supported by their original or primary vendor may be used if their original general availability date was within the last five years. The five-year limit is waived for hardware used in controller systems.

Software products that are still supported by their original or primary vendor may be used if their original general availability date was within the last three years.

If any component is no longer orderable by ordinary customers it must be disclosed in the FDR.

If a new or updated version of any software product is released causing earlier versions of said product to no longer be supported by the providing vendor(s), new publications or submissions occurring after four complete review cycles have elapsed must use a version of the product supported by the providing vendor(s).

For example, with result review cycles ending April 16, April 30th, May 14th, May 28th, June 11th, and June 25th, if a new JDK version released between April 16th and April 29th causes earlier versions of the JDK to no longer be supported by the providing vendor(s), results submitted or published on June 25th must use the new JDK version.

In addition, all components should conform to the clauses for general availability as described in the SPEC Open Systems Group Policies and Procedures document.

See http://www.spec.org/osg/policy.html#AppendixC – OSG Policy / Appendix C - Guidelines for General Availability

2.6.1. SUT Availability for Historical Systems

Please see OSG Policy section 2.3.5 on SUT Availability for Historical Systems http://www.spec.org/osg/policy.html#s2.3.5

See section 7.1 for SPECjbb2015-Composite, section 7.2 for SPECjbb2015-MultiJVM, and section 7.3 for SPECjbb2015-Distributed of this document for proper declaration of a historical model by listing “Historical” in the property“jbb2015.test.hwSystem=” in addition to system model information.

2.7. Configuration of System(s) Under Test (SUT)

The SUT may be a single stand-alone server or a multi-node set of servers as described below in the following sections. The SPECjbb2015 benchmark metrics apply to the entire SUT.

A multi-node SUT will consist of server nodes that cannot run independent of shared infrastructure such as a backplane, power-supplies, fans or other elements. These shared infrastructure systems are commonly known as “blade servers”.

Only identical servers are allowed in a multi-node SUT configuration; each must be identically configured. This requirement is for servers that execute the workload of the benchmark, and it does not include components that support these servers, e.g. storage-blades, controllers, and shared appliances.

All installed server-nodes must run the benchmark code, e.g. a multi-node SUT with 8 installed servers must run the benchmark code on all 8 nodes.

All software required to run the SPECjbb2015 benchmark must be installed on and executed from a stable storage device that is considered part of the SUT.

2.7.1. Compliant Run Configurations

Based on the hardware and software configuration, the SPECjbb2015 benchmark supports three different run categories. Users must submit results in an appropriate category as listed below for a given hardware and software configuration.

If the SUT runs a virtualized OS image, the Time Server must be used as outlined in the SPECjbb2015 User Guide under section 5.7 Running SPECjbb2015 with Time Server.

The SPECjbb2015 benchmark's three different run categories are:

1) SPECjbb2015-Composite: Single JVM / Single Host:

All the benchmark modules (Controller, Transaction Injector and Backend) run inside a single JVM process, in a single host. Use of clusters or aggregations of machines is specifically disallowed in this category. No network, database, or web server components are required, only a Java environment. In this category, the SUT consists of the host that runs the composite JVM process.

2) SPECjbb2015-MultiJVM: Multiple JVMs / Single Host:

The controller, each transaction injector and each backend must runs in separate JVM processes but all processes run in a single host. Use of clusters or aggregations of machines is specifically disallowed in this category; however network components may be required. There can be one or more Group(s) where one Group consists of one Backend mapped to one or more dedicated Transaction Injector(s). In this category, the SUT consists of the host that runs the benchmark JVM processes.

3) SPECjbb2015-Distributed: Distributed JVMs / Single or Multi Hosts.

The controller, transaction injector(s) and Backend(s) run in separate JVM processes. The controller and transaction injectors run on a separate host(s) than the Backend(s). These separate hosts are considered driver systems and are not part of the SUT. In this category, the SUT consists only of those hosts that run the Backend JVM processes.

To ensure accuracy of measurement timings, hosts running Controller and Transaction Injectors must run on non-virtualized environments. Controller and Transaction Injectors could be configured to run across multiple driver systems. For simplicity of description, all hosts running Controller and Transaction Injectors must be identically configured with the exception of JVM tuning parameters and instance count.

The Backend JVM processes are allowed to run on virtualized environments.

SUT configurations consisting of single system with multiple nodes must have the nodes identically configured.

SUT configurations consisting of multiple systems are only allowed if the system(s) are being productized and supported as a solution by a given vendor. These systems must be identically configured. The intent is to disallow multiple disparate systems publication.

More details can be found in the latest version of the design document on SPEC's website (http://www.spec.org/jbb2015/docs/designdocument.pdf).

2.7.2. Electrical Equivalence

Electrically equivalent submissions are allowed with the SPECjbb2015 benchmark.

2.7.3. Hardware

Any hardware configuration of one or more systems and supporting components that is sufficient to install, start, and run the benchmark to completion in compliance with these run rules (including the availability requirements in the General Availability section and multi-system requirements in section 2.4.2 under SPECjbb2015-Distributed category description) must be considered a compliant configuration. Any device configured at the time the

benchmark is started must remain configured for the duration of the benchmark run. Devices that are configured but not needed for the benchmark (e.g. additional on-board NICs) may be disabled prior to the start of the benchmark run.

External devices required for initial setup or maintenance of the SUT, but not required for normal operation or for running the benchmark (e.g. an external optical drive used for OS installation) may be removed prior to the benchmark being started.

If the model name or product number implies a specific hardware configuration, these specific components cannot be removed from the hardware configuration but may be upgraded. Any upgrades are subject to the support, availability and reporting requirements of this document. For example, if the SUT is available from the vendor only with dual power supplies, both supplies must be installed and measured during the benchmark run. The power supplies may be upgraded if the vendor offers and supports such an upgrade, and the upgrade must be documented in the benchmark disclosure report.

The components are required to be:

- Specified using customer-recognizable names,
- Documented and supported by the providing vendor, and
- Of production quality.

Any tuning or deviation from the default installation or configuration of hardware components is allowed by available tools only and must be reported. This includes BIOS settings, power saving options in the system board management, or upgrade of default components. Only modifications that are documented and supported by the vendor(s) are allowed.

2.7.4. Software

Required software components per server (host) are

- Exactly one single instance of a first level supervisor software, i.e. an operating system or a hypervisor hosting one or more instances of the same guest operating system. Each operating system instance includes all modules that are installed during the installation process and supports one user space.
- A Java runtime environment including one or more instances of a Java Virtual Machine (JVM).

The operating system must be in a state sufficient to execute a class of server applications larger than the benchmark alone. The majority of operating system services should remain enabled. Disabling operating system services may subject disclosures to additional scrutiny by the benchmark subcommittee and may cause the result to be found non-compliant. Any changes from the default state of the installed software must be disclosed in sufficient detail to enable the results to be reproduced. Examples of tuning information that must be documented include, but are not limited to:

- Description of System Tuning (includes any special OS parameters set, changes to standard daemons or services)
- List of Java parameters and flags used
- Any special per-JVM tuning for multi-JVM running (e.g. associating JVMs to specific processors)

These changes must be "generally available", i.e., available, supported and documented. For example, if a special tool is needed to change the OS state, it must be available to users, documented, and supported by the vendor.

The tester is expected to exercise due diligence regarding the reporting of tuning changes, to ensure that the disclosure correctly records the intended final product.

The software environment on the SUT is intended to be in a state where applications other than the benchmark could be supported. Disabling of operating system services is therefore discouraged but not explicitly prohibited. Disabled services must be disclosed.

The submitter/sponsor is responsible for justifying the disabling of service(s).

Services that must not be disabled include but are not limited to logging services such as cron or event logger.

A list of active operating system services may be required to be provided for SPEC's results review. The submitter is required to generate and keep this list for the duration of the review period. Such a list may be obtained, for example, by:

- Windows: net start
- Solaris 10: svcs -a
- Red Hat Linux: /sbin/runlevel; /sbin/chkconfig --list

2.8. Java Specifications

Tested systems must provide an environment suitable for running typical server-side Java SE 7.0 (or higher) applications. Any tested system must include an implementation of the Java (tm) Virtual Machine as described by the following references, or as amended by SPEC for later Java versions:

- Java Virtual Machine Specification (Second Edition / ISBN-13: 978-0201432947)

The following are specifically allowed, within the bounds of the Java Platform:

- Pre-compilation and on-disk storage of compiled executable binaries are specifically allowed. However, support for dynamic loading is required. Additional rules are defined in section 2.8.2. See section 2.6 for details about allowable flags for compilation.

The system must include a complete implementation of those classes that are referenced by this benchmark as in the Java SE 7.0 specification (http://www.oracle.com/technetwork/java/javase/downloads/index.html). SPEC does not intend to check for implementation of APIs not used in this benchmark.

For example, the benchmark does not use AWT (Abstract Window Toolkit), and SPEC does not intend to check for implementation of AWT. Note that the reporter does use AWT, however it is not necessary to run the reporter on the SUT.

2.8.1. Feedback Optimization and Pre-compilation

Feedback directed optimization and pre-compilation from the Java byte-code is allowed, subject to the restrictions regarding benchmark-specific optimizations in section 1.2.2. Pre-compilation and feedback-optimization before the measured invocation of the benchmark are also allowed. Such optimizations must be fully disclosed.

2.8.2. Benchmark Binaries and Recompilation

The SPECjbb2015 benchmark binaries are provided in jar files containing the Java classes. Valid runs must use the provided jar files and these files must not be updated or modified in any way. While the source code of the benchmark is provided for reference, the benchmark tester must not recompile any of the provided .java files. Any runs that use recompiled class files are marked invalid and cannot be reported or published.

2.8.3. Third Party libraries and packages

The benchmark is also using some third party libraries and packages. All such libraries and packages are being included with the benchmark. A user must not change or update to latest revisions, else, this will result in run being non-compliant and declared invalid. If there is a valid reason, a new kit and new version of the benchmark will be released.

3. Reporting Rules

In order to publicly disclose the SPECjbb2015 benchmark results, the tester must adhere to these reporting rules in addition to having followed the run rules above. The goal of the reporting rules is to ensure the system under test is sufficiently documented so that someone could reproduce the test and its results and to ensure that the tester has complied with the run rules.

3.1. Reporting Metric and Result

The SPECjbb2015 benchmark expresses performance in terms of two metrics, max-jOPS, which is a measure of the maximum throughput the SUT(s) can sustain without being subject to response time requirements; and critical-jOPS, which measures the maximum throughput the SUT(s) can sustain while being subject to response time requirements. Both the metrics should be displayed in proximity and comparison across the run categories is not allowed.

3.1.1. SPECjbb2015 benchmark Metric Names by Category

Category	Throughput metric name	Response time metric name
SPECjbb2015-Composite	SPECjbb2015-Composite max-jOPS	SPECjbb2015-Composite critical-jOPS
SPECjbb2015-MultiJVM	SPECjbb2015-MultiJVM max-jOPS	SPECjbb2015-MultiJVM critical-jOPS
SPECjbb2015-Distributed	SPECjbb2015-Distributed max-jOPS	SPECjbb2015-Distributed critical-jOPS

The report of results is an HTML file (*SPECjbb2015.wxyz-index.html*) generated by the tools provided by SPEC. These tools must not be changed, except for portability reasons with prior SPEC approval. The tools perform error checking and will flag some error conditions as resulting in an "invalid run". However, these automatic checks are only there for debugging convenience, and do not relieve the benchmark tester of the responsibility to check the results and follow the run and reporting rules.

Each benchmark run produces a binary data output file *SPECjbb2015-[C/M/D]-date-###.gz* which contains run details and must not be altered by the user. At the end of the run, benchmark reporter takes this binary data file and template raw file *template-[C/M/D].raw* containing SUT description as input and produces HTML report and *SPECjbb2015.wxyz.raw* file. *SPECjbb2015.wxyz.raw* file now has result summary along with SUT description. The section of the *SPECjbb2015.wxyz.raw* file that contains actual test measurement must not be altered. Corrections to the SUT descriptions may be made as needed to produce a properly documented disclosure.

3.1.2.

Organizations or individuals who makes public use of SPEC benchmark results must do so in accordance with the SPEC Fair Use Rule as posted at (http://www.spec.org/fairuse.html).

3.2. Required Disclosure for Independently Published Results

Please see SPEC OSG Rules: http://www.spec.org/osg/policy.html#s2.3.7

3.3. Reproducibility

SPEC is aware that performance results for production and pre-production systems may sometimes be subject to change, for example when a last-minute bug fix reduces the final performance. SPEC is also aware that the SPECjbb2015 benchmark contains non-deterministic functions that make results for repeated tests show a distribution of values so any single result is inherently an estimate of the true average value.

3.3.1. Pre-production system

If the sponsor becomes aware that for a typical released system the average max-jOPS or critical-jOPS metric is more than 5% lower than that reported for the pre-release system based on upper bound of 95% confidence interval determined with Student's t-test using at least 10 measured values from consecutive runs, the tester is required to submit a new result for the production system, and the original result must be marked non-compliant (NC).

3.3.2. Challenging a published result

A published result may be challenged as non-compliant by showing the average max-jOPS or critical-jOPS metric is more than 5% lower than that published result based on upper bound of 95% confidence interval determined with Student's t-test using at least 10 measured values from consecutive runs from a system configured and operated as disclosed in the published result.

In response, sponsor could show the validity of the published result by following the same procedure.

3.3.3. Examples

To compute whether the reported metrics are within the allowed range, take multiple measurements (benchmark runs) and determine the arithmetic mean and sample standard deviation of the entire set collected. The number of samples determines the degree of freedom (=N-1) to apply to the t-distribution with the desired confidence level to find the necessary coefficient to apply to the standard deviation divided by the square root of the number of samples.

Given a set of 11 benchmark runs that have a mean of 1000 and a sample standard deviation of 25, the degree of freedom is 10. From t Table lookup, the t coefficient for DF=10 with two-tails is 2.228 for the 95% confidence level.

2.228 * 25 / sqrt(11) = 16.79

The 95% confidence interval is the sample mean plus or minus the calculated value, 1000 +/- 16.79, or 983.21 (lower bound) to 1016.79 (upper bound). A published value for max-jOPS of greater than (1016.79 * 1.05) = 1067.54 would be challengeable.

Another example, given a set of 21 benchmark runs with the same mean and sample standard deviation would have a DF=20, so the 95% confidence level t coefficient is 2.086 (by lookup).

2.086 * 25 / sqrt(21) = 11.38

The 95% confidence interval is 1000 +/- 11.38, or 988.62 (lower bound) to 1011.38 (upper bound) and the limit for max-jOPS (1011.38 * 1.05) = 1064.61

3.4. Test-bed Configuration Disclosure

The following requirements apply to all hardware and software components used in producing the benchmark result, including the System Under Test (SUT), network, and controller. The system configuration information that is required to reproduce published performance results must be reported. The principle is that if anything affects performance or is required to duplicate the results, it must be described. Any deviations from the standard, default configuration for the SUT must be documented so an independent party would be able to reproduce the result without any further assistance.

For the following configuration details, there is an entry in the configuration file, and a corresponding entry in the tool-generated HTML result page. If information needs to be included that does not fit into these entries, the Notes sections must be used.

3.4.1. General Availability Dates

The dates of general customer availability must be listed for the hardware components and server software, by month and year. All the system, hardware and software features are required to be available within three months of the first publication of these results. With multiple sub-components of the major components having different availability dates, the latest availability date must be listed for that major component. The benchmark Software components are not included in this date.

3.4.2. Configuration Disclosure

Based on run category, the configuration file as described in User Guide Appendix A. The SPECjbb2015-Composite, SPECjbb2015-MultiJVM, and SPECjbb2015-Distributed templates must be filled out properly and all fields should be reflected in final benchmark report.

3.4.3. Benchmark Results Summary

The reporter automatically populates the Benchmark Result Summary. A graphical representation of these values is automatically rendered.

3.4.4. Validation and Runtime Checks

The following are required for a valid run and are automatically checked:

- The Java environment must pass the partial conformance testing.
- From the beginning of RT curve building phase to reaching max-jOPS, additional following run requirements must be met:
  - "Number of probes" validation criterion:
    - Transaction Injectors use probe requests to measure response time. To have good quality information for measuring response time there is a hard and soft limit criterion for the ratio of "# of probes jOPS / Total jOPS". A criterion is applied to each RT curve step level within required interval starting from the RT curve step level @0% to the last RT curve step level which meets 99th-percentile for response time for biggest SLA (100 ms).
      - Hard limit criteria states that the probes jOPS must be always 3% or more of the total number of jOPS
      - Soft limit criteria states that for 75% of the RT curve steps within required interval as defined above, the probe jOPS must be 5% or more of the total number of jOPS.
    - If this validation check is failing, increasing the thread pool size for issuing probes by setting property specjbb.customerDriver.threads.probe may help. This property should be greater or equal to 64 for compliance run. When increasing the number of probe workers you must also adjust the number of connections used to communication between TxInjector and Backend. The property specjbb.comm.connect.connpool.size specifies the number of connections and should be greater than (specjbb.customerDriver.threads.probe * 3) to avoid being a bottleneck. If you increase the number of connections it makes sense to increase the number of threads on backend side responsible for accepting request from TxInjector. This property is specjbb.comm.connect.pool.max and it should be greater than (specjbb.comm.connect.connpool.size * # of TxI for one backend + 10). Alternately, number of Transaction Injectors / Backend could also be increased using specjbb.txi.pergroup.count
    - If this criterion still fails, run is declared invalid.
  - "Request Mix Accuracy" validation criterion:
    - All requests being issued from Transaction Injectors maintains a specified % in the overall mix. For each RT curve step level within the interval starting from the RT curve step level @10% to the max-jOPS step level, following criteria is applied:
      - "actual % in the mix - expected % in the mix", must +/- 5% for all requests whose 'expected % in the mix as per design' is equal or more than 10% and must be +/- 1% for all requests whose 'expected % in the mix as per design' is less than 10%
      - If this criterion fails, run is declared invalid.
  - "Rate of Non-Critical Failures" validation criterion:
    - This criterion is applied to each RT curve step level within required interval starting from the RT curve step level @1% to the last RT curve step level which meets 99th-percentile for response time for biggest SLA (100 ms).
      - At each RT step level, "Rate of non-critical failures" must be less than 0.5% of IR (Injection Rate) at that RT step level.

        - If this criterion fails, run is declared invalid.
    - "Delay between performance ping during RT Curve" validation criterion:
        - The SPECjbb2015 benchmark Controller is sending requests to all agents every 1 sec and is receiving responses. The time elapsed between two subsequent responses received within the interval starting from the RT curve step level @0% to the max-jOPS step level, must pass a soft and hard limit criteria
            - As per soft limit criteria, no more than 10 times the time lapsed can exceed the 5 second for each agent.
            - Hard Limit criteria states that no time lapsed be more than 60 seconds.
            - If this criterion fails, run is declared invalid.

If there are any warnings in the run, the results can only be used after it has been reviewed and accepted for publication.

3.4.5. Validation and Runtime Checks when running with time server

For valid run with time server the time between SUT and the time server host should be well synchronized. Initial time offset at the beginning of the run between SUT and the time server host must not be greater than 10 minutes otherwise the run will not be valid.
When time server is enabled, Controller regularly collects the time stamps from SUT as well as from the timer server. Once run completes, SUT time offsets from the time server are identified for all the collected time stamps. Then the time offsets mean is calculated for the time stamps collected during Response-Throughput curve building phase. The following validation criteria are applied during Response-Throughput curve building phase:

- No more than 10 offsets where |mean - offset| > 50 msec
- No offset where |mean - offset| > 500 msec
- Offset STDDEV must not be > 100 msec

If the validation criteria fails then time on SUT is considered to be inaccurate and user need to resolve this issue.

4. Submission Requirements for the SPECjbb2015 benchmark

Result submissions for SPEC publication must include a single zip compressed package of the following two files:

1. The raw results file (SPECjbb2015-[C/M/D]-Date-runNum.raw)
2. A zip compressed file consisting of following files:
    - result/SPECjbb2015-[C/M/D]-Date-runNum/report-num/logs/SPECjbb2015-[C/M/D]-Date-runNumController.out
    - result/SPECjbb2015-[C/M/D]-Date-runNum/report-num/logs/SPECjbb2015-[C/M/D]-Date-runNumController.log
    - For SPECjbb2015-Distributed category, in addition to the above files, a pdf format diagram (Examples in section 7. Appendix A) showing SUT and driver systems details as well as network connectivity.

Overall package to be submitted will have a format:
Zip compressed package {*.raw + zip compressed files (*Controller.out + *Controller.log + when required *.pdf)}
where: *.raw file is used to generate the SPECjbb2015 benchmark HTML report and zip compressed files (*Controller.out + *Controller.log + when required *.pdf) is shown as a link in the SPECjbb2015 benchmark HTML report.

Once the submission is ready, please e-mail the required zip-compressed package (SPECjbb2015-[C/M/D]-Date-runNum.zip) to subjbb2015@spec.org

The committee may request additional benchmark output files from the submitter as well. The submitter should be prepared to participate in discussion during the review cycle and at the subcommittee meeting in which the result is voted on for final acceptance, to answer any questions raised about the result. The submitter is also required to keep the binary log files for the SUT and Controller for six months from the run. Licensees of the benchmark wishing to submit results for acceptance may be required to pay a fee. The complete submission process is documented in "Submitting OSG Benchmark Results to SPEC". (http://www.spec.org/osg/submitting_results.html).

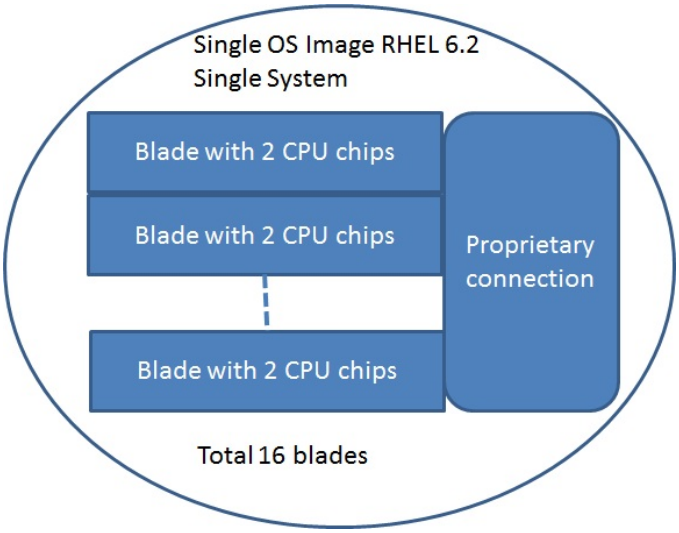
5. Trademark

SPEC and the name SPECjbb are registered trademarks of the Standard Performance Evaluation Corporation. Additional product and service names mentioned herein may be the trademarks of their respective owners.

6. Copyright Notice

7. Appendix A – Example FDR Package SUT Diagram in pdf format

7.1. SPECjbb2015-Composite and SPECjbb2015-MultiJVM



7.2. SPECjbb2015-Distributed