SPECpower Committee

Server Efficiency Rating Tool (SERT) TM

Design Document
3rd public draft

Standard Performance Evaluation Corporation

Table of Contents

1 Introduction 6
1.1 Summary 6
1.2 About SPEC 6
1.2.1 SPEC Membership 6
1.2.2 SPEC's General Development Guidelines 7
1.3 The EPA's ENERGY STAR for Computer Server Specification and SPEC 7
1.4 SERT's Differences from Conventional Benchmarks 7
1.5 Design Feedback Mechanism 8
1.6 Logistics 8
1.7 Trademark 8
2 Scope and Goals 9
2.1 Overview Summary 9
2.2 Sockets and Nodes 9
2.3 Scaling 9
2.4 Server Options and Expansion capabilities 10
2.5 IO Component 10
2.5.1 Storage IO 10
2.5.2 Network IO 10
2.6 Redundancy 11
2.7 Run Time 11
2.8 Platforms 11
2.8.1 Tested as Shipped 11
2.9 Implementation Languages 11
2.10 Load Levels 11
2.11 Worklets 12
2.12 Workload 12
2.13 Tentative Test Schedule 13
2.14 Schedule tradeoffs 13
3 SERT Architecture 14
3.1 System Overview 14
3.2 Execution of SERT 15
4 Worklet Execution Phases 16
5 Worklet Design Guidelines 19
5.1 Active Idle Worklet 19
5.2 CPU Worklet 19
5.3 Memory Worklet 19
5.4 Network IO Worklet 20
5.5 Storage IO Worklet 20
5.6 System Worklet 20
6 Power and Temperature Measurements 21
6.1 Environmental Conditions 21
6.2 Temperature Sensor Specifications 21
6.3 Power Analyzer Requirements 21
6.4 SPEC PTDaemon 22

6.5 Supported and Compliant Devices 22
6.6 Power Analyzer Setup 22
6.7 DC Line-Voltage 22
7 Metric/Score, Reporting, Logging 23
7.1 Metric/Score 23
7.2 Reporting and Output Files 23
7.2.1 Report 1: “Summary Report” 23
7.2.2 Report 2: “Power and Performance Data Sheet” 24
7.3 Validation / Verification 24
7.4 Logging 25
8 Future Enhancements / Stretch goals 26
8.1 Graphical User Interface (GUI) 26
8.2 Test Software 26
9 SERT and EPA ENERGY STAR for Server Version 2.0 27
9.1 Measurement 27
9.2 SERT Binaries and Recompilation 27
9.3 Manual Intervention 27
9.4 Fair Use of SERT information 27
9.4.1 Fair Use Rules 27
9.5 Accredited, Independent laboratory 27
9.6 Supply Voltage tolerance 28
10 Worklet Candidates 29
10.1 CPU Worklet: Compress 30
10.1.1 General Description 30
10.1.2 Sequence Execution Methods 30
10.1.3 Metric 30
10.1.4 Required Initialization 30
10.1.5 Configuration Parameters 30
10.1.6 Transaction Code 30
10.2 CPU Worklet: CryptoAES 31
10.2.1 General Description 31
10.2.2 Sequence Execution Methods 31
10.2.3 Metric 31
10.2.4 Required Initialization 31
10.2.5 Configuration Parameters 31
10.2.6 Transaction Code 31
10.3 CPU Worklet: FFT 32
10.3.1 General Description 32
10.3.2 Sequence Execution Methods 32
10.3.3 Metric 32
10.3.4 Required Initialization 32
10.3.5 Configuration Parameters 32
10.3.6 Transaction Code 32
10.4 CPU Workload: LU 33
10.4.1 General Description 33
10.4.2 Sequence Execution Methods 33

10.4.3 Metric ... 33
10.4.4 Required Initialization ... 33
10.4.5 Configuration Parameters ... 33
10.4.6 Transaction Code ... 33
10.5 CPU Workload: SOR ... 34
10.5.1 General Description ... 34
10.5.2 Sequence Execution Methods ... 34
10.5.3 Metric ... 34
10.5.4 Required Initialization ... 34
10.5.5 Configuration Parameters ... 34
10.5.6 Transaction Code ... 34
10.6 CPU Workload: XmlValidate ... 35
10.6.1 General Description ... 35
10.6.2 Sequence Execution Methods ... 35
10.6.3 Metric ... 35
10.6.4 Required Initialization ... 35
10.6.5 Configuration Parameters ... 35
10.6.6 Transaction Code ... 35
10.7 Memory Worklet: Flood ... 36
10.7.1 General Description ... 36
10.7.2 Sequence Execution Methods ... 36
10.7.3 Metric ... 36
10.7.4 Required Initialization ... 37
10.7.5 Configuration Parameters ... 37
10.7.6 Transaction Code ... 37
10.8 Memory Workload: XmlValidate ... 38
10.8.1 General Description ... 38
10.8.2 Sequence Execution Methods ... 38
10.8.3 Metric ... 38
10.8.4 Required Initialization ... 38
10.8.5 Configuration Parameters ... 38
10.8.6 Transaction Code ... 39
10.9 Storage IO Workload ... 40
10.9.1 General Description ... 40
10.9.2 Sequence Execution Methods ... 40
10.9.3 Metric ... 40
10.9.4 Required Initialization ... 40
10.9.5 Configuration Parameters ... 40
10.9.6 Transaction – Code 1 - RandomRead ... 41
10.9.7 Transaction – Code 1 - RandomWrite ... 41
10.9.8 Transaction – Code 2 – SequentialRead ... 41
10.9.9 Transaction – Code 2 – SequentialWrite ... 41
10.10 System Worklet: CSSJ ... 42
10.10.1 General Description ... 42
10.10.2 Sequence Execution Methods ... 42
10.10.3 Metric ... 42

10.10.4 Required Initialization 42
10.10.5 Configuration Parameters 42
10.10.6 New Order Transaction 42
10.10.7 Payment Transaction 43
10.10.8 Order Status Transaction 44
10.10.9 Delivery Transaction 44
10.10.10 Stock Level Transaction 44
10.10.11 Customer Report Transaction 44

1

1 Introduction

1.1 Summary

The EPA's ENERGY STAR development team is currently working on Version 2.0 of their Computer Server Specification[1]. Version 2.0 aims to evolve the program by adding a means to measure the overall efficiency of the server while it is performing actual computing work via an Active Mode Efficiency Rating Tool.

The SPECpower committee is currently working on the design, implementation and delivery of the Server Efficiency Rating Tool (SERT)™, a next generation tool set that will measure and evaluate the energy efficiency of computer servers. This public draft outlines the design of SERT for review by EPA stakeholders and their associates.

Please visit http://www.spec.org/sert/docs/SERT-Design_Doc.pdf for the latest updates.

1.2 About SPEC

The Standard Performance Evaluation Corporation (SPEC) was formed by the industry in 1988 to establish industry standards for measuring compute performance. SPEC has since become the largest and most influential benchmark consortium world-wide. Its mission is to ensure that the marketplace has a fair and useful set of metrics to analyze the newest generation of IT equipment.

The SPEC community has developed more than 30 industry-standard benchmarks for system performance evaluation in a variety of application areas and provided thousands of benchmark licenses to companies, resource centers, and educational institutions globally. Organizations using these benchmarks have published more than 20,000 peer-reviewed performance reports on SPEC's website (http://www.spec.org/results.html).

SPEC has a long history of designing, developing, and releasing industry-standard computer system performance benchmarks in a range of industry segments, plus peer-reviewing the results of benchmark runs. Performance benchmarking and the necessary work to develop and release new benchmarks can lead to disagreements among participants. Therefore, SPEC has developed an operating philosophy and range of normative behaviors that encourage cooperation and fairness amongst diverse and competitive organizations.

The increasing demand for energy-efficient IT Equipment has resulted in the need for power and performance benchmarks. In response, the SPEC community established SPECpower, an initiative to augment existing industry standard benchmarks with a power/energy measurement. Leading engineers and scientists in the fields of benchmark development and energy efficiency made a commitment to tackle this task. The development of the first industry-standard benchmark that measures the power and performance characteristics of server-class compute equipment started on January 26th 2006. In December of 2007, SPECpower_ssj2008 was released, which exercises the CPUs, caches, memory hierarchy and the scalability of shared memory processors on multiple load-levels. The benchmark runs on a wide variety of operating systems and hardware architectures. In version 1.10, which was released on April 15th 2009, SPEC augmented SPECpower_ssj2008 with multi-node support (e.g., blade-support).

1.2.1 SPEC Membership

SPEC membership is open to any interested company or entity. OSG members and associates are entitled to licensed copies of all released OSG benchmarks and unlimited publication of results on SPEC's public website. An initiation fee and annual fees are due for members. Nonprofit organizations and educational institutions have a reduced annual fee structure. Further details on membership information can be found on http://www.spec.org/osg/joining.html or requested at info@spec.org. Also a current list of SPEC members can be found here: http://www.spec.org/spec/membership.html.

[1] US Environmental Protection Agency – Energy Star Program Requirements for Computer Servers. http://www.energystar.gov/index.cfm?c=revisions.computer_servers

1.2.2 SPEC's General Development Guidelines

SPEC's philosophy and standards of participation are the basis for the development of SERT. The tool is being developed cooperatively by a committee representing diverse and competitive companies. The following guides the committee in the development of a tool that will be useful and widely adopted by the industry:

- Decisions are reached by consensus. Motions require a qualified majority to carry.
- Decisions are based on reality. Experimental results carry more weight than opinions. Data and demonstration overrule assertion.
- Fair benchmarks allow competition among all industry participants in a transparent market.
- Tools and benchmarks should be architecture-neutral and portable.
- All who are willing to contribute may participate. Wide availability of results on the range of available solutions allows the end user to determine the appropriate IT equipment.

Similar guidelines have resulted in the success and wide use of SPEC benchmarks in the performance and power/performance industry and are essential to the success of SERT.

1.3 The EPA's ENERGY STAR for Computer Server Specification and SPEC

SPEC applauds the EPA for its goal to drive toward greater energy efficiency in IT Equipment, and SPEC considers the EPA ENERGY STAR Program an industry partner in this effort. The development of an Active Mode Efficiency Rating Tool is an essential component in the ongoing effort to reduce worldwide energy consumption and paves the way for a successful ENERGY STAR for Computer Servers program that has the potential to harmonize energy efficiency programs worldwide.

SPEC welcomes this opportunity to work with the EPA on SERT in support of the ENERGY STAR Specification for Computer Server and is proudly looking forward to continuing our long-standing association with the EPA ENERGY STAR development team.

1.4 SERT's Differences from Conventional Benchmarks

Performance benchmarks and energy efficiency benchmarks tend to focus on capabilities of computer servers in specific business models or application areas. SERT is focused on providing a first order of approximation[2] of energy efficiency across a broad range of application environments.

- The absolute score is less relevant for the end user, because it will not reflect specific application capabilities.
- A rating tool that provides a pass-fail or a [Level 1/Level 2/Level 3] pass-fail rating is a better fit for EPA's ENERGY STAR Environment for Computer Servers than a typical benchmark result with multiple digits of precision in the metric.
- Marketing of the absolute scores will be disallowed, in order to encourage more participation in the program

Benchmarks tend to focus on optimal conditions, including tuning options to customize the configuration and software to the application of the benchmark business model. The need to achieve competitive benchmark results often causes significant investment in the benchmark process. SERT is designed to be more economical and easier to use, requiring minimal equipment and skills through:

- Highly automated processes and leveraging existing SPEC methods
- Focus on as-shipped default settings for the server
- Free from super-tuning

Where a benchmark represents a fixed reference point, ENERGY STAR programs are designed to foster continuous improvement, with thresholds for success rising as the industry progresses. SERT will be designed to match this paradigm, including:

- Quick adoption of new computing technologies
- Rapid turn-around for SERT version updates

[2] Andrew Fanara, Evan Haines, Arthur Howard
http://www.energystar.gov/ia/partners/prod_development/downloads/State_of_Energy_and_Performance_Benchmarking_for_Enterprise_Servers_Final.pdf

1.5 Design Feedback Mechanism

The SERT development team will evaluate input from a broad spectrum of industry experts during the entire development process by utilizing its partnership with the EPA ENERGY STAR Program. The team will collaborate on workload, metric and all other requirements of the EPA's Version 2.0 Framework.

Please provide your detailed feedback to the EPA via servers@energystar.gov. The EPA will collect, sort, anonymize, and prioritize your feedback and pass it on to the SPEC development team.

1.6 Logistics

The licensee and price structure as well as the support and maintenance models that will be used for SERT is work in progress.

1.7 Trademark

Product and service names mentioned herein may be the trademarks of their respective owners.

2 Scope and Goals

The current scope of Version 2.0 ENERGY STAR for Computer Servers includes servers with 1-4 processor sockets with a stated goal to expand to include blade technologies of similar scope. A design goal of SERT is to accommodate these and larger technologies.

Among the issues involved with support of larger systems are the overall capacity of the system to complete work, and the ability to design a workload that scales with the inclusion of additional processors, memory, network interface cards, disk drives, etc. Different workload characteristics are required to demonstrate effectiveness for each of these components. Providing a workload that fairly represents their presence while not unfairly representing their absence is a challenge. These issues are more prevalent with larger systems that have more expansion capabilities than smaller servers.

For these areas where it is concluded that the tool does not adequately represent the value of a component compared to its power requirements, the tool will be designed to accommodate the inclusion of “configuration power/performance modifiers”. A design goal is to automatically include this additional information in the computation of the ENERGY STAR qualification results, including detailed documentation that this was done.

2.1 Overview Summary

The following table summarizes some of the design goals that SERT will and will not provide.

IS	IS NOT
Rating Tool for overall energy efficiency	A Benchmark nor a Capacity Planning Tool
Measuring tool for power, performance and inlet-temperature	Measuring tool for Airflow, Air pressure, outlet-temperature
General compute-environment measure	Specific application benchmark measure
Support of AC- powered servers	Support of DC-powered servers
Used in single OS instance per server environments	Intended to stress virtualization hypervisor technology [3]
ENERGY STAR Rating Tool	Marketing Tool
Planned to be architecture and OS neutral	Planned to be implemented on architecture and/or OS environments where insufficient resource has been volunteered to accomplish development, testing, and support.

2.2 Sockets and Nodes

SERT 1.0.0.0 is designed to be scalable and will be tested up to a maximum of 8 sockets and a maximum of 64 nodes (limited to a set of homogenous servers or blade servers). The server under test (SUT) may be a single stand-alone server or a multi-node set of servers. 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” or “multi-node server”. Only identical servers are allowed in a multi-node SUT configuration.

2.3 Scaling

Since the server efficiency rating of a given server is the primary objective of SERT, one of the main design goals for SERT is to be able to scale the performance on the system in proportion to the system configuration. As more components (processors, memory, and disk storage) are added to the server, the workloads should utilize the additional resources so that the resultant performance is higher when compared to the performance on the same server with a lesser configuration. Similarly, for a given server, when the components are upgraded with faster counterparts, the performance should scale accordingly. This is a very important aspect of the tool since adding and upgrading components typically increases the total power consumed by the server which will affect the overall efficiency result of the server. Creating a tool that scales performance based on the number/speed of CPUs is most readily achievable – for the other components, the complexity of implementing such a tool increases substantially.

[3] Virtualization can be an important tool for saving energy. In a first-order approximation tool, such as SERT, the impacts of virtualized environments can be determined by examining the results at higher load levels.

While SERT will be designed to scale performance with additional hardware resources of the SUT, if there are performance bottlenecks in system components unrelated to the added hardware the SUT itself may not be able to sustain higher performance. In such cases the addition of components to the SUT will normally result in higher power consumption without a commensurate increase in performance. It is also possible that the workload mix that is defined for smaller systems will not scale well when examining larger systems.

2.4 Server Options and Expansion capabilities

A server may have many optional features that are designed to increase the breadth of applications. These features not only absorb additional power, but also require more capacity in the power supplies and cooling system. Some SERT workload components will be designed to demonstrate the enhanced capabilities that these features provide. However, while the tool needs to credit these capabilities for the expanded workloads that they will accommodate, it cannot penalize efficient servers that are not designed with substantial expansion options. A balance must be struck between providing enhanced ratings for enhanced configurations and avoiding easy qualification of servers by simply adding features that may not be needed in all situations.

SERT's goal is to avoid unnecessarily penalizing servers that are designed for low expandability, while crediting servers with greater expandability. For example a configuration with four I/O adapters in PCI slots may execute the workload of the tool more effectively than a configuration with only one such adapter. On the other hand it may only run the workload of the tool as effectively as a configuration with two network adapters. Because the configuration with four adapters may run some real workloads more effectively than configurations with only two adapters, the EPA may elect to allow for some form of "configuration modifier" to provide credit for the power infrastructure needed to support the additional PCI slots.

The tool will be designed and tested to ensure that, should "configuration power-performance modifier" credits be included, the tool will accommodate them.

2.5 IO Component

Disk and Network IO components are strongly desired to provide a better-rounded picture of system performance and power than a CPU-centric test. SPEC is in the early stages of evaluating IO workloads for SERT, so this section provides many discussion points but not necessarily conclusions.

SPEC recognizes that some of the items in the next two sections may not be reasonable or practical to test or measure in a meaningful way. In those cases we would suggest the use of "configuration power-performance modifiers" to compensate for the extra power draw associated with extra functionality. Other items under consideration include:

- Different types/quantities of IO for different server categories
- Self-calibrating performance measurements for the disk and network subsystem

2.5.1 Storage IO

Ideally the Storage IO component of SERT would give credit for:

- Higher performance storage subsystems
- Larger capacity storage subsystems
- Reliability and availability features (RAID, battery backed cache, etc)

2.5.2 Network IO

Ideally the network IO component of SERT would give credit for:

- Higher performance Network Interfaces
- Larger transfer speed Network Interfaces
- Reliability and availability features

2.6 Redundancy

Many servers have redundancy built in for power supplies and cooling fans. Some servers include different levels of redundancy for memory, disk, and even processors. A design goal is to include accommodation for redundant components, although no specific tests are planned for energy measurement under fault tolerant conditions when one of a redundant set of components is disabled.

2.7 Run Time

The right balance between high repeatability of the results, high sub-system coverage and low resource allocation is desirable. The run time will depend on the agreed set of worklets. The target run time is around 3 hours.

2.8 Platforms

SERT 1.0.0.0 will be implemented for and is planned to be tested on the following platform/OS/JVM combinations (64 bit only), pending resources. In some cases, SPEC recommend the use of more than one JVM, where more than one JVM is generally available and selecting one may unfairly penalize a specific processor architecture or operating system.

HW Platform	x86 AMD	x86 AMD	x86 AMD	x86 Intel	x86 Intel	x86 Intel	Itanium Intel	POWER IBM	POWER IBM	POWER IBM	SPARC Oracle	SPARC Fujitsu
OS	Windows Server 2008 R2	LINUX	Solaris	Windows Server 2008 R2	LINUX	Solaris	HP-UX 11i	AIX	IBM i	LINUX	Solaris	Solaris
JVM	IBM j9 Oracle HS	IBM j9 Oracle HS	Oracle	IBM j9 Oracle HS	IBM j9 Oracle HS	Oracle	HP HS	IBM – j9	IBM – j9	IBM –j9	Oracle	Oracle

Note: OS refers to versions (service pack and patch levels) that are current at the SERT release.

Platform/OS/JVM combinations currently not on the list have no resources allocated to them. If support for additional architectures or OSs is desired, then active participation from requesting entities is mandatory. The inclusion of a JVM is dependent on an agreement from the JVM provider for unrestricted use of their JVM for SERT. Companies dedicating additional resources to the SPECpower committee for development of SERT would relax the schedule constraints.

2.8.1 Tested as Shipped

To provide results that are representative of a customer environment, the goal is to test systems in an “as-shipped” state. No super tuning would be allowed, but rather a limited list of valid parameter changes for configuration and typical optimization be permitted. Other changes will cause the run to be marked as noncompliant. SERT will launch the JVM within the tool, to restrict additional tuning.

The list of allowable parameters will be included in a future version of this document and in the operational documentation of the tool. This list would be agreed with the EPA before SERT release, and would be clearly documented as part of the SERT Run Rules.

2.9 Implementation Languages

The main body of code in written in Java in order to lower the burden of cross-platform support. Regardless, the framework is designed to accommodate other language implementations as well.

2.10 Load Levels

Multiple load levels are a desired goal of SERT and the design will include support for multiple levels. The active idle load level as well as a 100% workload level (not max power) are already good candidates. Prototype testing will show which levels will be included and if any weighting will be necessary.

2.11 Worklets

Developing the workload in the traditional SPEC way based on real world applications would result in complex test environments and high run times, especially for the IO intensive workloads, e.g. many client systems would be required for network IO and large disk sub systems for storage IO. The resulting costs for running such tests could be prohibitive for a rating tool. Therefore the SERT workload will be a collection of synthetic worklets for a variety of different load scenarios.

2.12 Workload

The existing SPEC benchmarks are mainly based on tailored versions of real world applications representing a typical workload for one application area or a synthetic workload derived from the analysis of existing server implementations. These benchmarks are suitable to evaluate different sub-areas of the overall server performance or efficiency if power measurements are included. They are not designed to give a representative assessment of the overall server performance or efficiency.

The design goal for the SERT suite however is to include all major aspects of server architecture, thus avoiding any preference for specific architectural features which might make a server look good under one workload and show disadvantages with another workload.

The SERT workload will instead take advantage of different server capabilities by using various load patterns, which are intended to stress all major components of a server uniformly.

If some components cannot be stressed adequately by the respective load pattern this can be compensated by adjusting the threshold for these components, e.g. increasing the power allowance for additional components which are not used by the load pattern.

It is highly unlikely that a single workload can be designed which achieves the goals outlined above, especially given the time constraints of the schedule targeted for ENERGY STAR for Servers Version 2.0 by the EPA. Therefore the SERT workload will consist of several different worklets each stressing specific capabilities of a server. This approach furthermore supports generating individual efficiency scores for the server components besides the overall system score.

Figure 1 describes the general structure of the SERT test suite and its components.

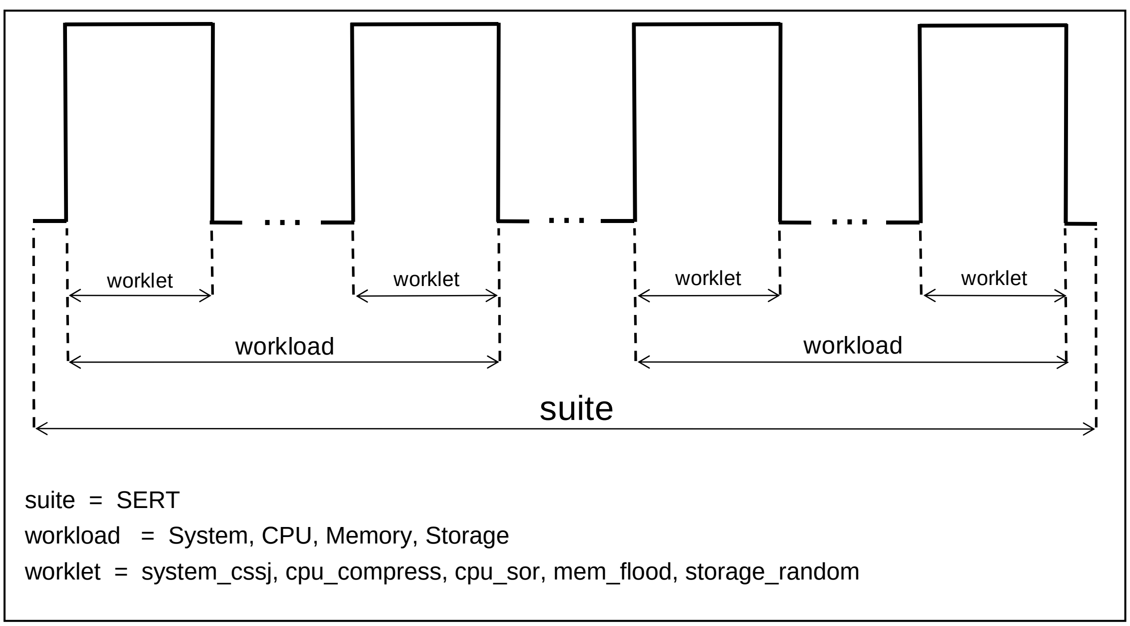


Figure 1: SERT Suite Components

2.13 Tentative Test Schedule

The alpha test phase is planned to start in March 2011 and the start of each phase requires successful completion of its predecessor. An estimated schedule can be created once we have decided on all design details.

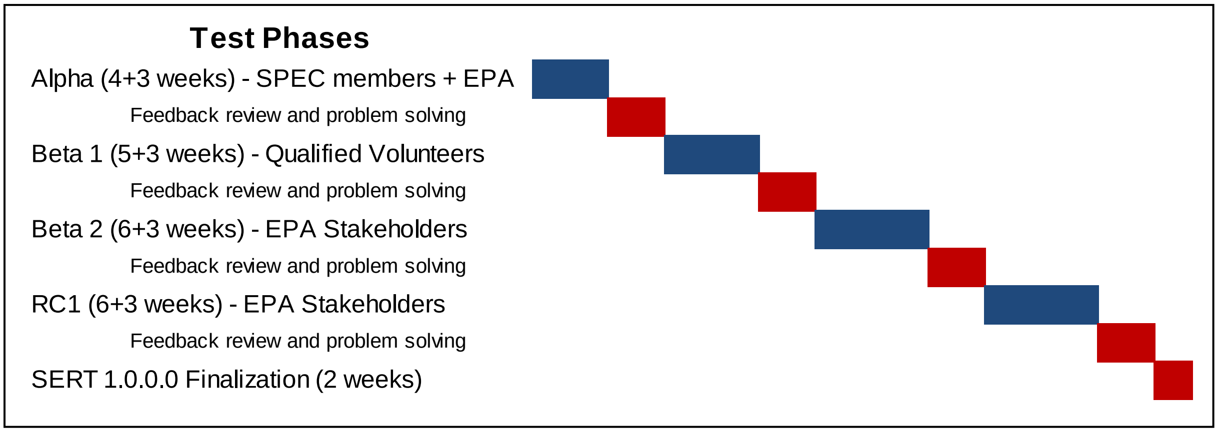


Figure 2: Tentative Test Schedule

2.14 Schedule tradeoffs

SPEC benchmarks are developed with the goal to generate results which are directly comparable for multiple hardware and software architectures to the extent this is possible. The same basic goal directs the design of SERT as specified in this document.

Even though SERT is designed with the goal of being architecture agnostic, code needs to be implemented for each of the workloads and the tool harness on all supported architectures. Furthermore this code must be tested intensively on all architectures in order to ensure a functionally equivalent set of binaries, which generate fair and comparable results. Simply using a portable programming language will not be sufficient to achieve these goals. Consequently significant complexity is added to the development process.

Given that SERT is designed as a first order approximation rating tool, comparability may be handled differently than with benchmarks (second order approximation tools) which are used for competitive marketing. Nevertheless it’s essential to ensure a minimal level of comparability.

The resources available in the SPECpower committee are limited and a timely development of the tool for a single architecture will be challenging. Support for additional architectures will remove resources from the development of the basic test routines because they will be needed for porting the code. Furthermore additive testing effort is required not only for the new architectures but for the original implementation as well in order to ensure comparability. Therefore each extra architecture will add a currently undetermined amount of time to the schedule. The resource and schedule problems recur with the support of multiple operating systems. SERT will be initially implemented on selected Operating Systems (OS) per HW architecture.

3 SERT Architecture

3.1 System Overview

SERT shares design philosophies and elements from SPECpower_ssj2008 in its overall architecture. SERT is composed out of multiple software components.

For the most basic SERT hardware measurement setup one of each of the following is required:

- system under test (SUT) – the actual system for which the measurements are being taken. The controller and SUT are connected to each other via an Ethernet connection.
- controller (e.g. server, PC, laptop) – the system to which the power analyzer and temperature sensor are connected.
- power analyzer – connected to the controller and used to measure the power consumption of the SUT.
- temperature sensor – connected to the controller and used to measure the ambient temperature where the SUT is located.

The SERT is composed of several elements including:

- the test harness (Chauffeur) – handles the logistical side of measuring and recording power data along with controlling the software installed on the SUT and controller system itself.
- the director – instructs the SUT to execute the workload.
- the workload (a set of worklets) – exercises the SUT while the test harness collects the power and temperature data.
- the SPEC PTDaemon – connects to the power analyzer and temperature sensor and gathers their readings while the workload executes.
- the reporter – gathers the environmental, power and performance data after a run is complete and compiles it into an easy to read format.
- Future versions of the kit will also include a GUI to ease setting up and executing the kit.

The basic system overview diagram shows these components in relationship to each other.

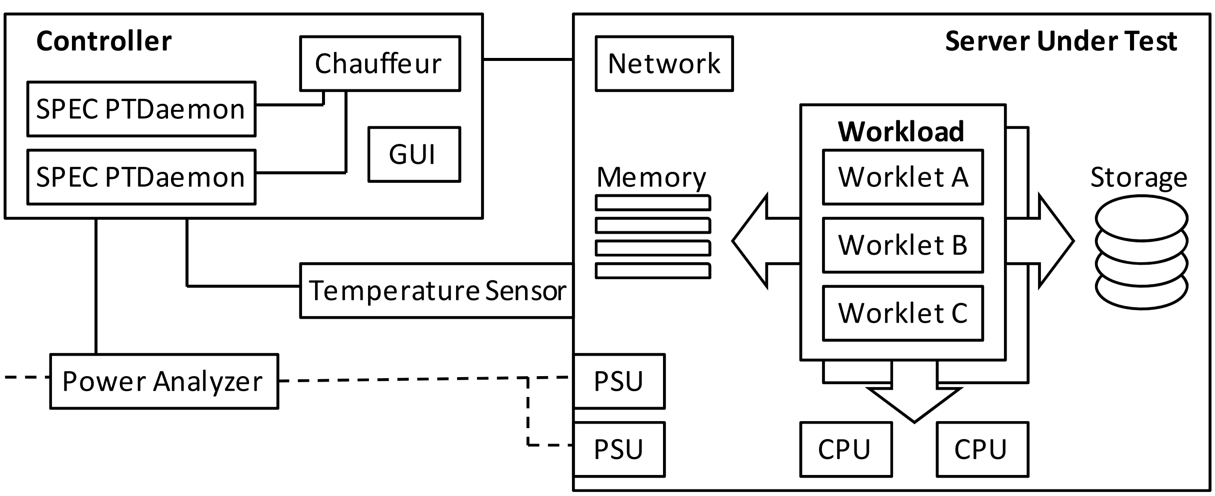


Figure 3: SERT Overview

3.2 Execution of SERT

These basic steps are needed in order to execute the SERT kit:

1. Setup the power analyzer and its associated SPEC PTDaemon script.
    - Configure the power analyzer to correctly measure the amperage and voltage of the SUT.
    - Edit the runpower.bat/sh script file to ensure that the proper power analyzer model is specified and the correct communication and network ports are used.
    - Ensure that the SPEC PTDaemon connects and communicates with the power analyzer.
2. Setup the temperature sensor and its associated SPEC PTDaemon script.
    - Edit the runtemp.bat/sh script file to ensure the proper temperature sensor model is specified and the correct communication and network ports are used.
    - Ensure that the SPEC PTDaemon connects and communicates with the temperature sensor.
3. Edit the Director script file.
    - Edit the director.bat/sh script file for the appropriate system configuration.
    - Ensure the proper Java path is specified.
    - Ensure the LOCAL_DIRECTOR variable contains the appropriate information.
    - Ensure the DIRECTOR_HOST variable contains the appropriate information.
4. Edit the SERT script file.
    - Edit the sert.bat/sh script file for the appropriate system configuration.
    - Ensure the proper Java path is specified.
    - Ensure the proper number of JVM's is specified.
    - Ensure the LOCAL_DIRECTOR variable contains the appropriate information.
    - Ensure the DIRECTOR_HOST variable contains the appropriate information.
5. Run the SPEC PTDaemon, Director and SERT scripts.
    - Execute the runpower.bat/sh, runtemp.bat/sh, director.bat/sh and sert.bat/sh scripts.

After the kit completes the run, there should be a results.xml file located in the \results\chauffeur-xxxx directory (where xxxx is the run iteration number).

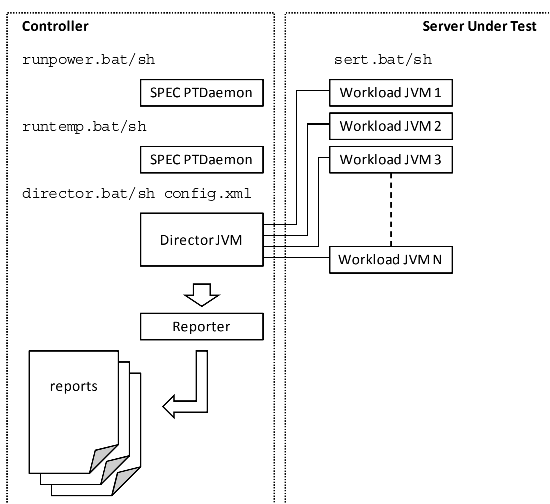


Figure 4: SERT Startup Procedure

4 Worklet Execution Phases

The SERT test suite consists of several workloads, which are designed to stress the various components of the system under test (SUT), e.g. CPU, Memory, Storage. Each workload includes one or more worklets, which execute specific code sequences to load one of these system components. The overall design structure of SERT is shown in Figure 1 above.

In the current design the worklets will run consecutively, each in its own phase. This allows generating independent scores at different load levels which can be combined to an overall metric.

Selected worklets or additional worklets specifically designed for concurrent execution may be run simultaneously in an extra phase. Concurrent execution of worklets will introduce more realistic task-switching, especially using IO load modules.

The worklet execution is split into phases called “sequences” in SERT terminology. Currently the SERT harness supports two types of worklet sequences, “Graduated Measurement” and “Fixed Iteration”.

Those worklets, which can be throttled to lower load levels will use the graduated measurement execution scheme described in Figure 5 Worklets, which always run at maximum capacity have to use the fixed iteration execution scheme shown in Figure 6.

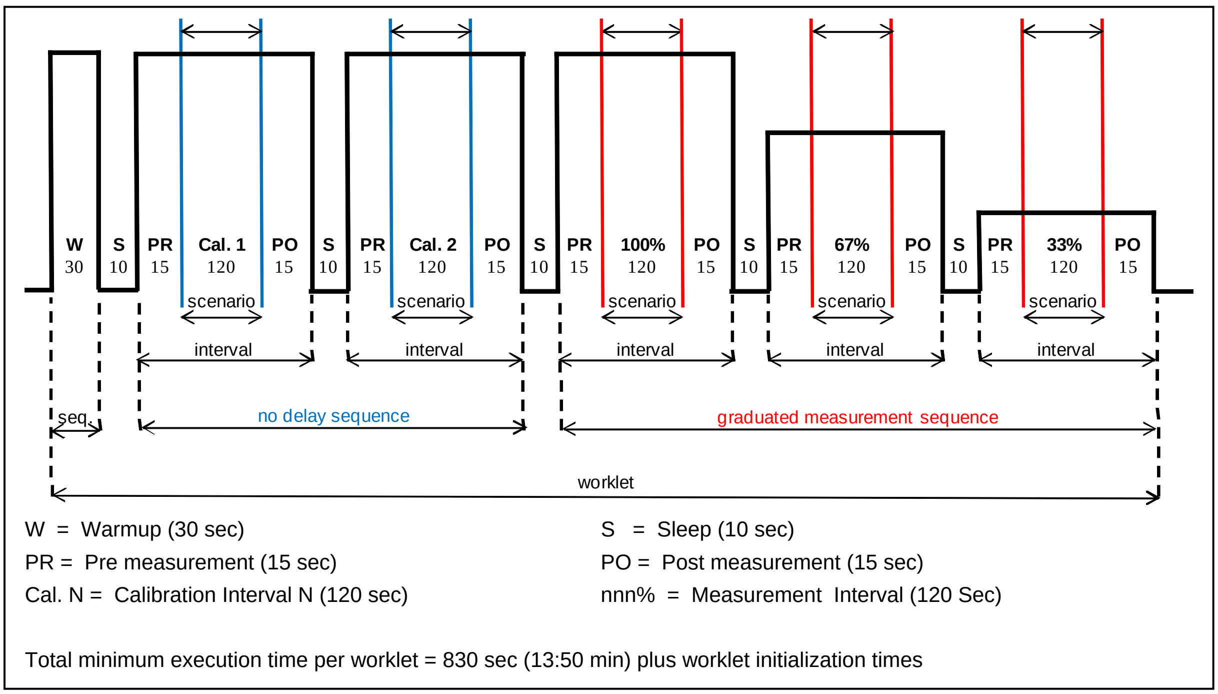


Figure 5: Graduated Measurement Execution

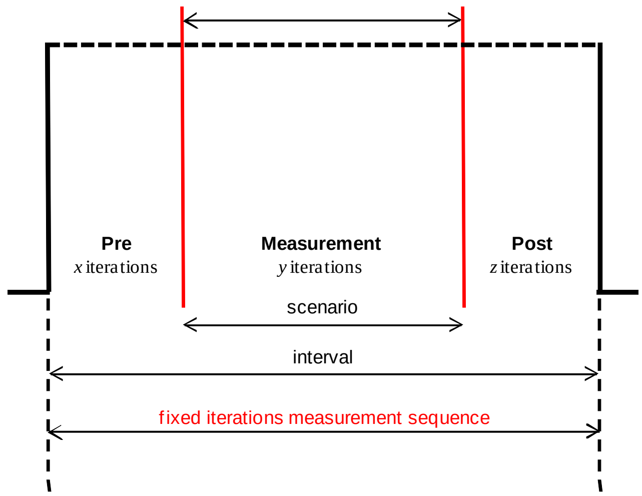
The following series of phases is typically performed for graduated measurement execution, interrupted by short sleep phases:

1. Warmup Sequence
2. No Delay Sequence for calibration
3. Graduated Measurement Sequence for determining the performance and power data

For the warmup and calibration phases the worklet code is executed unrestricted, whereas the SERT harness controls the load levels during the measurement sequence.

Each sequence can be divided into intervals of fixed time length as shown in Figure 5. The number of intervals, their duration and the desired load levels can be defined in SERT configuration files individually for each worklet.

Examples of worklets using graduated measurement execution are System_CSSJ, CPU_Compress, Storage_Random.



Pre = Pre measurement (x iterations) (optional)

Post = Post measurement (z iterations) (optional)

Measurement = Measurement (y iterations)

Total minimum execution time per worklet = not fixed, depends on system capacity

Figure 6: Fixed Iteration Execution

The fixed iteration execution scheme typically includes one sequence and one interval only. The duration of the interval is not predefined but determined by the capacity of the system, i.e. the time it takes to execute the fixed amount of work.

The number of intervals and scenarios can be defined in SERT configuration files for each worklet individually.

Currently only the Mem_Flood worklet uses this execution scheme

User	A User is a representation of a external agent that can initiate work (e.g. human being)
	Each User may maintain identifying information
	E.g. each User represents a Warehouse
	Each User may maintain state information
	Temporary information that persists from one transaction to another
	There may be multiple types of Users for a single Workload
Transaction	A transaction receives a User and transaction-specific Input as parameters
	It produces some Result
	Some transactions may be able to verify their results – this could be used for a small portion of transactions for auditing purposes
Scenario	A worklet is a set of transactions that can be executed by a particular type of User
	Workloads may contain multiple worklets
	Each worklet could represent a sequence of user interactions
	Think time may occur between transactions
Interval	Each interval in a sequence includes pre-measurement
	Within each interval, each User schedules the execution of worklets
	When a scenario's scheduled time arrives, it iterates through its transactions. Each transaction is submitted to a JVM-wide thread pool. The next transaction in the Worklet will be submitted after the current transaction completes

Sequence	Each phase consists of a sequence of intervals	
	The intervals in a sequence have something in common (though the “something” can vary based on workload or configuration)	
	no delay sequence	fixed time intervals running scenarios unrestricted
	graduated measurement sequence	fixed time intervals with controlled execution of scenarios
	fixed iterations measurement sequence	a predefined number of iterations per scenario is executed, the score is calculated from the execution time, the pre and post interval phases are optional and may be missing for some worklets
Worklet	A workload defines a set of Users and worklets	
	Execution of a Workload includes multiple phases: Warmup Calibration One or more measurement phases	
	Each of these phases are really a sequence of measurement intervals	
	Multiple measurement phases could be used for varying transaction mix, users, etc	

5 Worklet Design Guidelines

In order to achieve consistent results from all worklets and a broad coverage of technologies the following guidelines should be observed:

- Each worklet must be adjustable to different performance levels, e.g. some predefined levels between 100% (maximum load) and 0% (idle)
- Each worklet must calibrate to maximum performance level by itself, i.e. no definition of the 100% level by the test user
- Multiple programming languages may be used
- Precompiled binaries of the test programs should be used where possible.
- Each worklet should scale with the available hardware resources. More resources should result in a higher performance score, e.g. more processor/memory/disk capacity or additional processor/memory/disk modules yield a better result in the performance component of the efficiency rating.
- Portable code that follows all SPEC rules for licensing, reuse and adaptation.
- Either architecture and OS agnostic or with "if-def" capability to accommodate different architectures and/or OSs.
- The work accomplished by each worklet is clearly identifiable as "important" but is not required to cover "all important" types of work.

In order to follow these guidelines the workloads will probably be based on batches of discrete work, where each batch constitutes a transaction. The different load levels will be achieved by scheduling the required number of transactions.

5.1 Active Idle Worklet

During active idle measurements, the SUT must be in a state in which it is capable of completing workload transactions. The active idle worklet is treated in a manner consistent with all other worklets, with the exception that no transactions occur during the active idle interval.

5.2 CPU Worklet

A combination of a wide variety of processor-intensive tasks, including string manipulation, task management, Java "commercial" processing, C "commercial" processing, numeric processing, and other tasks as identified and appropriate.

- Consistent processor characteristics per simulated "user" regardless of number of processors, cores, enabled threads, etc.
- Bottleneck at 100% is the processor, not the storage or memory
- Able to schedule processor tasks or blocks of tasks in such a way that the load can be scaled from 100% in graduated levels down to idle.
- The CPU worklets should measure a higher (better) performance score for:
    - higher #CPU, higher #core, higher #logical processors, higher frequency, larger overall cache, lower latency, faster interconnect between CPU sockets

5.3 Memory Worklet

Combination of random and sequential reads and writes, small and large memory accesses.

- Consistent memory access characteristics per simulated "user" regardless of size and number of memory DIMMs
- Bottleneck at 100% is the memory itself, not the processor or storage
- Able to schedule memory stress tasks or blocks of tasks in such a way that the load can be scaled from 100% in graduated levels down to idle.
- The memory worklets should measure a higher (better) performance score based on memory characteristics (e.g. higher bandwidth, lower latency, total memory size)

5.4 Network IO Worklet

Configuration power/performance modifier will be established in order to address Network IO.

- Avoid expensive and extensive external test system configurations
- Measurements show that there are no significant differences in power utilization between 100% and 0% network utilization for today's technology

5.5 Storage IO Worklet

Combination of random and sequential, reads and writes, small and large I/Os.

- Consistent I/O characteristics per simulated "user" regardless of system size and number of disks or the installed memory
- Bottleneck at 100% is the storage subsystem, not the processor or memory
- Able to schedule I/O tasks or blocks of tasks in such a way that the load can be scaled from 100% in graduated levels down to idle.
- The storage worklets should measure a higher (better) performance score for a higher bandwidth and lower latency

The measurements of power and performance of either optional add-in storage controller cards or server blade enclosure storage are not in the scope of SERT.

5.6 System Worklet

A combination of a wide variety of processor and memory-intensive tasks

- Bottleneck at 100% is the processor and memory
- Able to schedule processor tasks or blocks of tasks in such a way that the load can be scaled from 100% in graduated levels down to idle.
- The system worklets should measure a higher (better) performance score for:
    - higher #CPU, higher #core, higher #logical processors, higher frequency, larger overall cache, lower latency, faster interconnect between CPU sockets
    - higher bandwidth, lower latency, total memory size

6 Power and Temperature Measurements

SERT provides the ability to automatically gather measurement data from accepted power analyzers and temperature sensors and integrate that data into the SERT result. It will be required that the analyzers and sensors must be supported by the measurement framework, and be compliant with the specifications in this section.

6.1 Environmental Conditions

Power measurements need to be taken in an environment representative of the majority of usage environments. The intent is to discourage extreme environments that may artificially impact power consumption or performance of the server, before and during the SERT run.

The following environmental conditions need to be met:

- Ambient temperature lower limit: 20°C
- Ambient temperature upper limit: within documented operating specification of SUT
- Elevation: within documented operating specification of SUT
- Humidity: within documented operating specification of SUT
- Overtly directing air flow in the vicinity of the measured equipment in a way that would be inconsistent with normal data center practices is not allowed.

6.2 Temperature Sensor Specifications

Temperature must be measured no more than 50mm in front of (upwind of) the main airflow inlet of the SUT. To ensure comparability and repeatability of temperature measurements, SPEC requires the following attributes for the temperature measurement device used during the benchmark:

- Logging - The sensor must have an interface that allows its measurements to be read by the benchmark harness. The reading rate supported by the sensor must be at least 4 samples per minute.
- Accuracy - Measurements must be reported by the sensor with an overall accuracy of +/- 0.5 degrees Celsius or better for the ranges measured during the benchmark run.

6.3 Power Analyzer Requirements

To ensure comparability and repeatability of power measurements, the following attributes for the power measurement device are required for SERT. Please note that a power analyzer may meet these requirements when used in some power ranges but not in others, due to the dynamic nature of power analyzer Accuracy and Crest Factor. The usage of power analyzer's auto-ranging function is not permitted.

- Measurements - the analyzer must report true RMS power (watts) and at least two of the following measurement units: voltage, amperes and power factor
- Accuracy - Measurements must be reported by the analyzer with an overall uncertainty of 1% or better for the ranges measured during the benchmark run. Overall uncertainty means the sum of all specified analyzer uncertainties for the measurements made during the benchmark run.
- Calibration - the analyzer must be able to be calibrated by a standard traceable to NIST (U.S.A.) (http://nist.gov) or a counterpart national metrology institute in other countries. The analyzer must have been calibrated within the past year.
- Crest Factor - The analyzer must provide a current crest factor of a minimum value of 3. For Analyzers which do not specify the crest factor, the analyzer must be capable of measuring an amperage spike of at least 3 times the maximum amperage measured during any 1-second sample of the benchmark run.
- Logging - The analyzer must have an interface that allows its measurements to be read by the SPEC PTDaemon. The reading rate supported by the analyzer must be at least 1 set of measurements per second, where set is defined as watts and at least 2 of the following readings: volts, amps and power factor. The data averaging interval of the analyzer must be either 1 (preferred) or 2 times the reading interval. "Data averaging interval" is defined as the time period

over which all samples captured by the high-speed sampling electronics of the analyzer are averaged to provide the measurement set.

Examples:

An analyzer with a vendor-specified accuracy of +/- 0.5% of reading +/- 4 digits, used in a test with a maximum power value of 200W, would have "overall" accuracy of (((0.5%*200W)+0.4W)=1.4W/200W) or 0.7% at 200W.

An analyzer with a wattage range 20-400W, with a vendor-specified accuracy of +/- 0.25% of range +/- 4 digits, used in a test with a maximum power value of 200W, would have "overall" accuracy of (((0.25%*400W)+0.4W)=1.4W/200W) or 0.7% at 200W.

6.4 SPEC PTDaemon

SPEC PTDaemon (also known as power/temperature daemon, PTD or ptd) is used by SERT to offload the work of controlling a power analyzer or temperature sensor during measurement intervals to a system other than the SUT. It hides the details of different power analyzer interface protocols and behaviors from the SERT software, presenting a common TCP-IP-based interface that can be readily integrated into different benchmark harnesses.

The SERT harness connects to PTDaemon by opening a TCP port and using the simple commands detailed in the API section of this document. For larger configurations, multiple IP/port combinations can be used to control multiple devices.

PTDaemon can connect to multiple analyzer and sensor types, via protocols and interfaces specific to each device type. The device type is specified by a parameter passed locally on the command line on initial invocation of the daemon.

The communication protocol between the SUT and PTDaemon does not change regardless of device type. This allows SERT to be developed independently of the device types to be supported.

6.5 Supported and Compliant Devices

SERT will utilize SPEC's accepted measurement devices list and SPEC PTDaemon update process. See Device List (http://www.spec.org/power_ssj2008/docs/device-list.html) for a list of currently supported (by the SPEC PTDaemon) and compliant (in specifications) power analyzers and temperature sensors.

6.6 Power Analyzer Setup

The power analyzer must be located between the AC Line Voltage Source and the SUT. No other active components are allowed between the AC Line Voltage Source and the SUT.

Power analyzer configuration settings that are set by the SPEC PTDaemon must not be manually overridden.

6.7 DC Line-Voltage

SPEC PTDaemon is neither supported nor tested with DC loads today and currently no resources are devoted to including this support. We are in favor of including DC support if new resources from companies whose focus is DC computing become available to the SPECpower committee to address the development and support opportunity.

Additional, comparing servers powered by AC against servers powered by DC is not fair, since the AC-DC conversion losses are not included in DC-powered server. Therefore we recommend creating a separate category for DC-powered servers.

7 Metric/Score, Reporting, Logging

7.1 Metric/Score

While SERT is not intended to be a benchmark, nevertheless as a rating tool it must produce a metric or score indicative of the efficiency of the server under test. That metric must combine both the performance of the SUT as well as its power consumption in a way that allows comparison among all systems subjected to it. The desired outcome of that comparison is a quantitative measure of the relative power-performance efficiencies of the systems. The system which produces the higher metric should have greater power-performance efficiency than the system which produces the lower metric.

Since different architectures perform differently on different workloads, SERT is composed of several discrete worklets to ensure architecture neutrality. Each worklet will produce a measure representing the performance achieved by the SUT, which then must be combined with the measures produced by the other worklets to yield a metric indicative of the overall performance of the SUT on all worklets used in the tool. SPEC recommends that the multiple performance measures produced in this manner be combined into a single metric as the geometric mean of the individual measures.

The geometric mean of individual worklet performance may be used whether the individual worklets are run sequentially or simultaneously. Depending on the worklets chosen and the magnitudes of their individual measures, we intend indexing the measures to a set of reference scores before combining them into the single metric as the geometric mean. These techniques have the advantages of rendering the single metric unit-less, and of keeping the scale of the individual measures within similar ranges, so that a worklet with large magnitude individual measure does not overwhelm the result from a workload with a smaller measure.

Once determined, the overall performance must be combined with the measured power consumption of the SUT in a way that demonstrates the power-performance efficiency of the system. This will be a complex calculation automatically performed by SERT to take into account the power-performance efficiency of the SUT at different utilization levels.

The metric that is produced by SERT is separate from the ENERGY STAR rating. The EPA will determine criteria for ENERGY STAR acceptance of which the SERT scores may be only a part. It's anticipated that the top 25% of tested units will achieve ENERGY STAR qualification. A “gold-level” ENERGY STAR qualification may be available for units achieving in the top 5% of results. Additionally the EU has proposed a system of graduated achievement in power-performance efficiency with levels A through F, for which they will determine the overall criteria.

Server under test may be placed in different categories by the EPA. The EPA will decide how to apply these categories and whether units in a particular category may be compared to units in another category.

7.2 Reporting and Output Files

SERT will produce two reports and a set of log files. The reports will be created in XML format, in order to reduce the effort for both EPA and the partner in displaying and or storing the desired information. We will take steps in order to ensure authenticity (e.g. encryption) of the reports.

7.2.1 Report 1: “Summary Report”

This report will contain a placeholder for a “pass or fail” notice for the tested platform, to be provided by the EPA. A test run is marked non-compliant if the test completes with technical errors. In such a case, error messages and/or warnings will be automatically included in the report. The information in this report is public and could be used for marketing purpose.

Items included in this report are:

- EPA Partner name and EPA Partner ID
- EPA ENERGY STAR Category of the tested platform
- Test Date and Location (plus “Tested by”)
- Tested Platform Manufacturer and Model Number
- Placeholder for “Pass/Fail”

- Warnings or Error Notices if applicable
- System Configuration information (Redundant components to be marked appropriately):
  - form factor
  - number and type of processors
  - available processor sockets
  - memory size, type, # memory DIMMs, # DIMM Slots, Max Memory Capacity
  - available expansion slots
  - number of and make-model of power supply, output rating, min/max
  - Input power
  - OS supported / OS used for test
  - number of and make-model of storage controller
  - number of and make-model of mass storage devices
  - number of and make-model of network interface cards (NICs)
  - Management Controller or Service Processor Installed? [Yes/No]
  - Other Hardware Features / Accessories

7.2.2 Report 2: "Power and Performance Data Sheet".

This report will contain all the information the EPA requires and that is deemed necessary by SPEC. The Power and Performance Data Sheet will be public, but marketing use is prohibited by SERT Fair Usage Rules. The information is intended to be delivered to the EPA in a form most expeditious for EPA review.

This report will contain all the data from the "Summary Report" with the following additional detail sections:

- Overall Result / Score
- All target load level results
- Hardware and Software Configuration
- Power Measurement Summary
- Environmental information

7.3 Validation / Verification

SERT software components will implement software checks wherever possible to increase information accuracy, verify user input, monitor run-time data collection, and validate results with the intent of improving accuracy and remedying user errors, preventing invalid reports to the EPA.

When conditions or results do not meet specific criteria, warnings will be displayed and error messages will appear in the SERT reports.

These features will make it easy for the EPA Partner to generate compliant results and prevent submission of erroneous reports to the EPA.

Examples of compliance checking are:

- Verify input properties (parameters) and run-time duration of load levels.
- Temperature out of range will be reported.
- Power and Temperature read errors must be under a chosen threshold.

All the SERT software components will perform validation checks within the domain of their functions, e.g. warnings of connection problems, log measurement errors and out-of-range conditions, warning the user of missing or incomplete information and check the validity of some entered data.

Other new validation methods will be considered as the SERT software design and implementation progresses.

7.4 Logging

A set of log files will be produced for each test run.

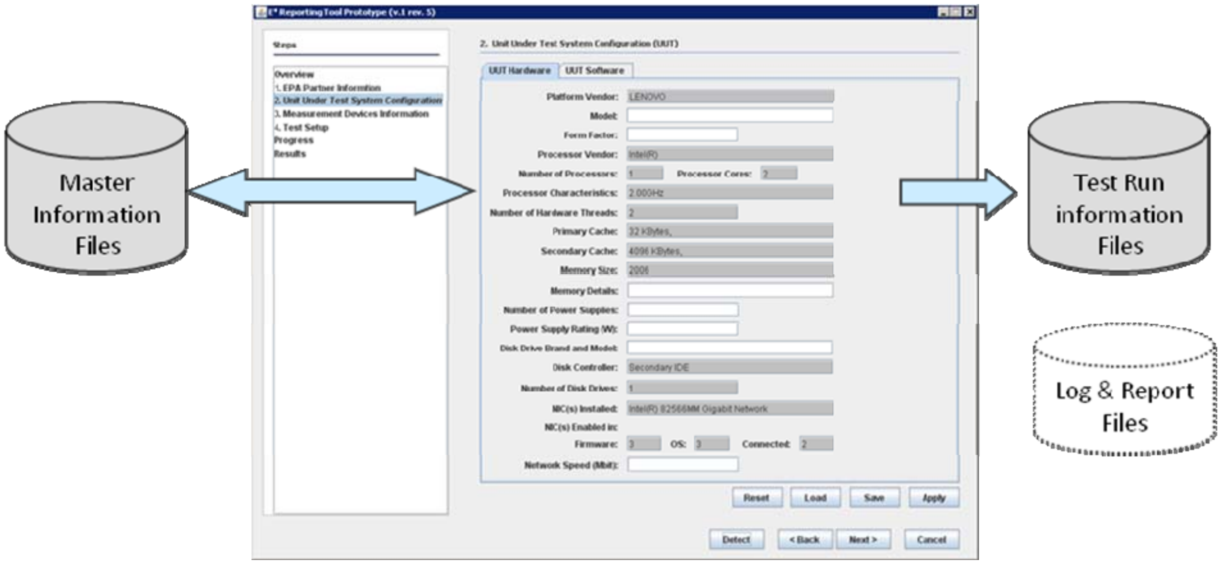
- The information in the log files is intended to be “non-public”.
- These files will be identified by a run serial number such that multiple consecutive test runs produce multiple log file sets.
- Each log file will be a record of actions from the software during the various phases of the testing, including errors and warnings.
- The intent of the log files is for auditing and support purpose.
    - Problems or failures can be more easily resolved with this low level detail record. If any issues arise with regard to the accuracy or veracity of the partner reports, these log files (potentially encrypted) should be adequate to resolve most issues.
- Examples of log file content are:
    - Handshake validation messages among various components
    - Error or warning messages
    - State change messages/notifications.
    - ‘Transaction’ instantaneous/periodic summary information
    - ‘Transaction’ response times

The EPA may require that any or all of the above outputs be delivered prior to ENERGY STAR qualification. Regardless, the partner must commit to archiving all output from any results submitted to the EPA.

8 Future Enhancements / Stretch goals

8.1 Graphical User Interface (GUI)

One of the stretch goals is the incorporation of a graphical user interface (GUI) to facilitate configuration and setup of test runs, allow real-time monitoring of test runs and to review the results. The SERT GUI will lead the user through the steps of detecting or entering the hardware and software configuration, setting up a trial run or a valid test, displaying results reports and other functions common to the testing environment.



The SERT GUI will include several features to enable SERT testing with minimal training and enhance the accuracy of results:

- Easy Navigation with Tabbed Screens
- How to Use (in-line usage guidance and help)
- Configuration Discovery (Detect function) will automatically populate most fields about SUT and Controller hardware and software.
- The GUI will display, allow entry of and store required information about the test environment
    - For use in reports: e.g. Company Info, Platform Config, Run-Time parameters, etc.
    - Master and Test Run information files can be stored, enabling reuse, saving time with multiple platforms.
- Test Setup, Execution and Progress Display
    - Start measurements; Choose type of run (trial or final)
    - Display progress, warnings and errors.
- Display results and enable printing and capture of reports
- Provisions for redundant components and power and performance modifier.

8.2 Test Software

A “stretch goal” of SERT is to enable a “Live CD” approach to tool installation, for some environments – such that the entire tool suite along with the underlying operating system could all be run from a single bootable CD or DVD with no other operating system installed on the SUT. This should provide increased ease of installation and improve the adoption rate of the tool.

Possible issues with this approach include the lack of specific hardware drivers for newer devices, the potential lack of vendor specific power management, licensing and availability issues for some operating systems. Alternatives include allowing additional drivers to be installed during setup, or providing separate test installers with binaries for use with a vendor’s own as-shipped OS installation.

9 SERT and EPA ENERGY STAR for Server Version 2.0

In order to ensure that SERT is utilized in the intended matter, we recommend the inclusion of the following items in the ENERGY STAR for Server Specification.

9.1 Measurement

The provided SERT test kit must be used to run and produce measured SERT results. The SERT metric is a function of the SERT workload (see section 0). SERT results are not comparable to power and performance metrics from any other application.

9.2 SERT Binaries and Recompilation

Valid runs must use the provided binary files and these files must not be updated or modified in any way.

9.3 Manual Intervention

No manual intervention or optimization for the SUT or its internal and external environment is allowed during the test measurement, after initial setup is completed.

9.4 Fair Use of SERT information

A clear goal of the ENERGY STAR program is to have the broadest possible participation among vendors. Experience in the computer industry’s performance benchmark community demonstrates that when performance details become available for marketing purposes, only vendors with superior (at the time of publication) products are incented to publish results. To encourage broader participation across the industry, a set of strong rules must be in place that will restrict marketing use of any of the detailed information generated by the tool. No data besides the actual ENERGY STAR qualification should be utilized in EPA Partners’ marketing collateral. These rules will be stipulated in both the license for the tool and the EPA Partner agreement.

Note that, while these rules are not strictly a part of the tool “design”, the existence of these rules are necessary to allow the flexibility of the design and the delivery of detailed consumer information that is desired.

9.4.1 Fair Use Rules

- The only information provided by the tool that can be used for marketing collateral is the ENERGY STAR qualification of a server configuration or server family
- The only information provided by the tool that can be used for public comparison is the ENERGY STAR qualification of a server configuration or server family All other publicly available information from the tool is made available to help to verify that the tests were run correctly and to allow consumers to better understand how well the configurations tested match their specific needs.
- If the tool is used for research to generate information outside of the ENERGY STAR program, the information may not be compared to the ENERGY STAR program results and competitive comparisons may not be made using the data generated.
- The EPA ENERGY STAR Qualification is governed by EPA rules.

9.5 Accredited, Independent laboratory

The requirement to use accredited, independent laboratories may place a large burden on EPA ENERGY STAR partners, especially smaller companies. We recommend the use of an independent laboratory as an option, but not implementing this as a requirement.

9.6 Supply Voltage tolerance

In order to use a voltage within a 1% difference, an extra voltage source is needed. This will unnecessarily increase the cost for the partner, especially smaller companies. We recommend the tolerance be set to ± 5%.

10 Worklet Candidates

The following table shows the current Worklet candidates and their anticipated use in different SERT test phases. Worklet candidates included in early releases may change in subsequent releases. Early release test results may influence the inclusion of some worklets in future releases.

Workload	Worklet candidate	Alpha	Beta 1	Beta 2	RC1
CPU	CPU_Compress	Included	TBD	TBD	TBD
CPU	CPU_CryptoAES	Included	TBD	TBD	TBD
CPU	CPU_SOR	Included	TBD	TBD	TBD
CPU	CPU_FFT	Included	TBD	TBD	TBD
CPU	CPU_LU	Included	TBD	TBD	TBD
CPU	CPU_XMLvalidate	Included	TBD	TBD	TBD
Memory	Mem_Flood	Included	TBD	TBD	TBD
Memory	Mem_XMLvalidate1	Included	TBD	TBD	TBD
Memory	Mem_XMLvalidate2	Included	TBD	TBD	TBD
Storage	Storage_Random	-	TBD	TBD	TBD
Storage	Storage_Sequential	-	TBD	TBD	TBD
Storage	Storage_Mixed	Included	TBD	TBD	TBD
System	System_CSSJ	Included	TBD	TBD	TBD

10.1 CPU Worklet: Compress

10.1.1 General Description

The Compress workload implements a transaction that compresses and decompresses data using a modified Lempel-Ziv method (LZW). Essentially, it finds common substrings and replaces them with a variable size code. This is both deterministic and done on the fly. Thus, the decompression procedure needs no input table, but tracks the way the table was built. The algorithm is based on "A Technique for High Performance Data Compression", Terry A. Welch, IEEE Computer Vol 17, No 6 (June 1984), pp 8-19.

10.1.2 Sequence Execution Methods

Graduated Measurement Sequence

10.1.3 Metric

Transactions Per Second

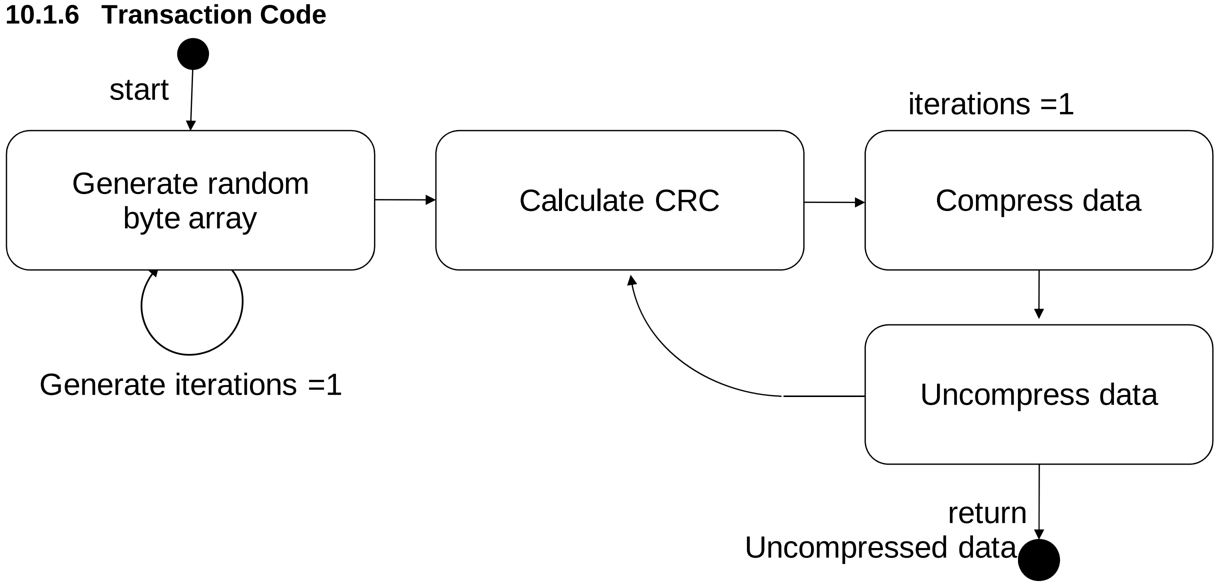
10.1.4 Required Initialization

A constant size byte array is generated on the fly before for each transaction execution. The contents of the byte array are randomly generated.

10.1.5 Configuration Parameters

`size`	Size of the input byte array for each transaction execution.
`enable-idc`	Enables/disables memory scaling using input data caching (IDC). Must be set to false.
`iterations`	Number of executions per transaction.
`debug-level`	Value governs the volume of debug messages printed during execution.
`input-generate-iterations`	Number of random byte array assignment iterations.

10.1.6 Transaction Code



10.2 CPU Worklet: CryptoAES

10.2.1 General Description

The CryptoAES workload implements a transaction that encrypts and decrypts data using the AES (or DES) block cipher algorithms. Which algorithm is a configurable parameter, but the current candidate version uses AES with CBC and no PKCS5 padding. Encryption and decryption are done using the Java Cryptographic Extension (JCE) framework, and the Cipher class in particular.

10.2.2 Sequence Execution Methods

Graduated Measurement Sequence

10.2.3 Metric

Transactions Per Second

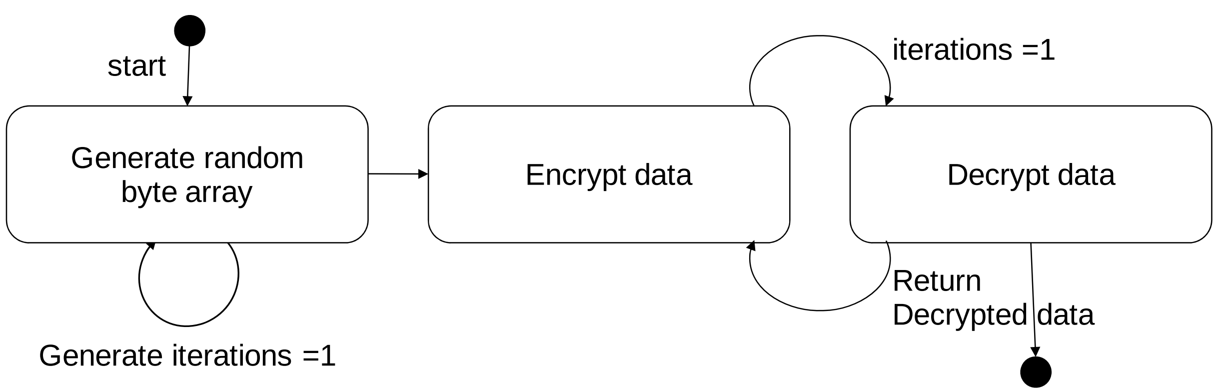
10.2.4 Required Initialization

A constant size byte array is generated on the fly before for each transaction execution. The contents of the byte array are randomly generated.

10.2.5 Configuration Parameters

`size`	Size of the input byte array for each transaction execution.
`key-generator`	Key generator algorithm. (AES or DESede)
`key-size`	Key size. (128 for AES, 168 for DES)
`algorithm`	Encryption algorithm. (E.g., AES/CBC/NoPadding, AES/CBC/PKCS5Padding, DESede/CBC/NoPadding, DES/CBC/PKCS5Padding)
`level`	Number of times to perform the encryption.
`enable-idc`	Enables/disables memory scaling using input data caching (IDC). Must be set to false.
`iterations`	Number of executions per transaction.
`debug-level`	Value governs the volume of debug messages printed during execution.
`input-generate-iterations`	Number of random byte array assignment iterations.

10.2.6 Transaction Code



10.3 CPU Worklet: FFT

10.3.1 General Description

The Fast Fourier Transform (FFT) workload implements a transaction that performs a one-dimensional forward transform of complex numbers. Its floating point computations exercise complex arithmetic, shuffling, non-constant memory references and trigonometric functions. The first section performs the bit-reversal portion (no flops) and the second performs the actual Nlog(N) computational steps. (Adapted from the NIST-developed Scimark benchmark.)

10.3.2 Sequence Execution Methods

Graduated Measurement Sequence

10.3.3 Metric

Transactions Per Second

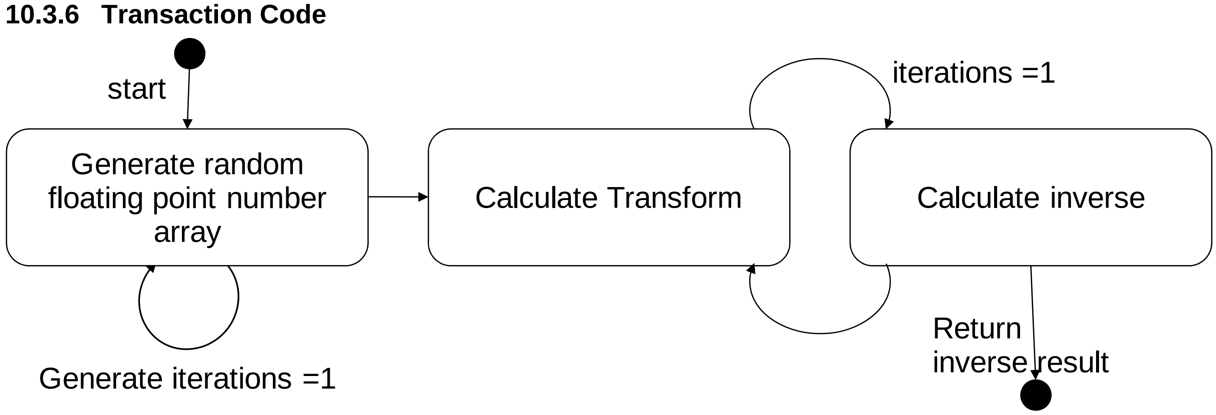
10.3.4 Required Initialization

A constant size floating point number array is generated on the fly before for each transaction execution. The contents of the array are randomly generated.

10.3.5 Configuration Parameters

`array-length`	Size of the input floating point number array for each transaction execution.
`enable-idc`	Enables/disables memory scaling using input data caching (IDC). Must be set to false.
`iterations`	Number of executions per transaction.
`debug-level`	Value governs the volume of debug messages printed during execution.
`input-generate-iterations`	Number of random array assignment iterations.

10.3.6 Transaction Code



10.4 CPU Workload: LU

10.4.1 General Description

The LU workload implements a transaction that computes the LU factorization of a dense matrix using partial pivoting. It exercises linear algebra kernels (BLAS) and dense matrix operations. The algorithm is the right-looking version of LU with rank-1 updates. (Adapted from the NIST-developed Scimark benchmark.)

10.4.2 Sequence Execution Methods

Graduated Measurement Sequence

10.4.3 Metric

Transactions Per Second

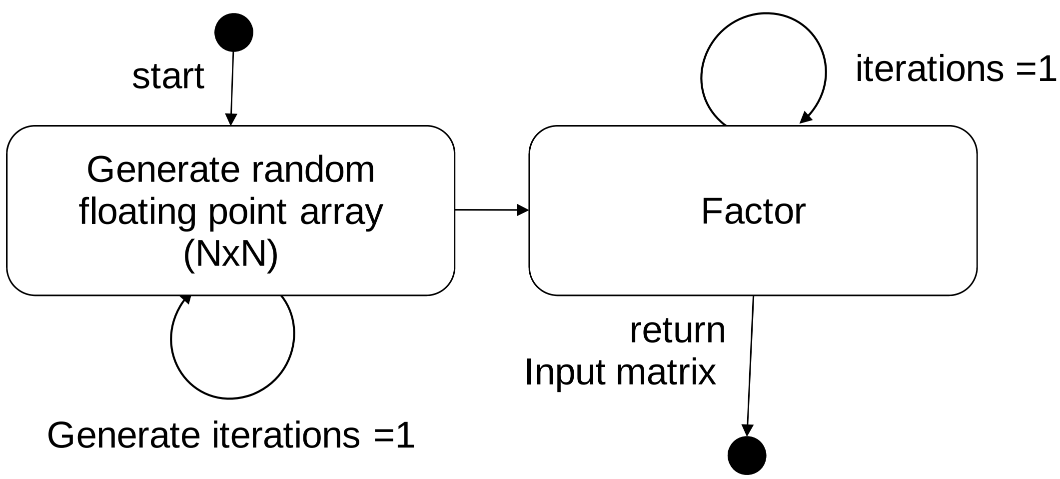
10.4.4 Required Initialization

A constant size matrix of floating point numbers is generated on the fly before for each transaction execution. The contents of the matrix are randomly generated.

10.4.5 Configuration Parameters

`matrix-dimen`	Dimension of the input floating point matrix for each transaction execution. (NxN)
`enable-idc`	Enables/disables memory scaling using input data caching (IDC). Must be set to false.
`iterations`	Number of executions per transaction.
`debug-level`	Value governs the volume of debug messages printed during execution.
`input-generate-iterations`	Number of random matrix assignment iterations.

10.4.6 Transaction Code



10.5 CPU Workload: SOR

10.5.1 General Description

The Jacobi Successive Over-relaxation (SOR) workload implements a transaction that exercises typical access patterns in finite difference applications, for example, solving Laplace's equation in 2D with Drichlet boundary conditions. The algorithm excercises basic "grid averaging" memory patterns, where each *A(i,j)* is assigned an average weighting of its four nearest neighbors. Some hand-optimizing is done by aliasing the rows of G[][] to streamline the array accesses in the update expression. (Adapted from the NIST-developed Scimark benchmark.)

10.5.2 Sequence Execution Methods

Graduated Measurement Sequence

10.5.3 Metric

Transactions Per Second

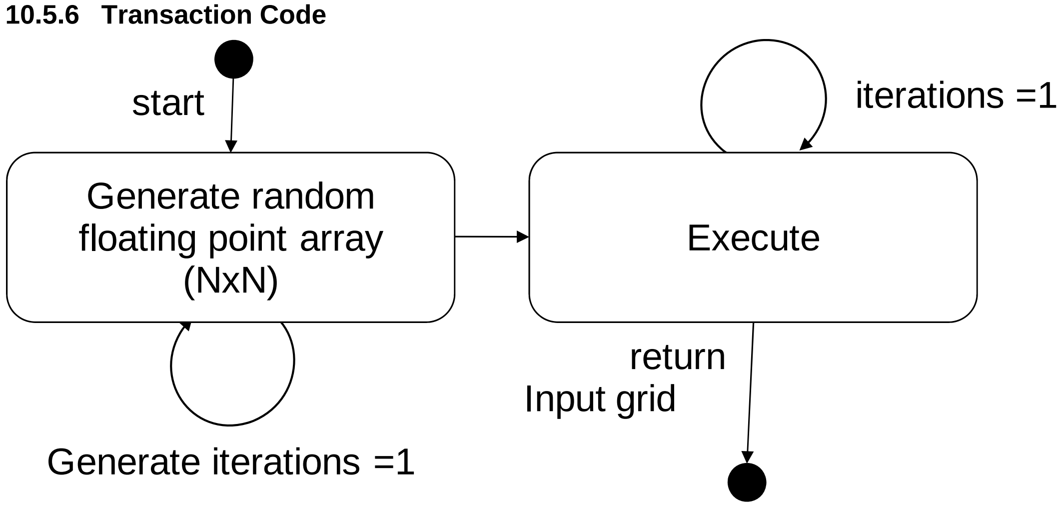
10.5.4 Required Initialization

A constant size grid of floating point numbers is generated on the fly before for each transaction execution. The contents of the grid are randomly generated.

10.5.5 Configuration Parameters

`grid-dimen`	Dimension of the input floating point grid for each transaction execution. (NxN)
`enable-idc`	Enables/disables memory scaling using input data caching (IDC). Must be set to false.
`iterations`	Number of executions per transaction.
`debug-level`	Value governs the volume of debug messages printed during execution.
`input-generate-iterations`	Number of random grid assignment iterations.

10.5.6 Transaction Code



10.6 CPU Workload: XmlValidate

10.6.1 General Description

The XML validate workload implements a transaction that exercises Java's XML validation package javax.xml.validation. Using both SAX and DOM APIs, an XML file (.xml) is validated against an XML schemata file (.xsd). To randomize input data, an algorithm is applied that swaps the position of commented regions within the XML input data.

10.6.2 Sequence Execution Methods

Graduated Measurement Sequence

10.6.3 Metric

Transactions Per Second

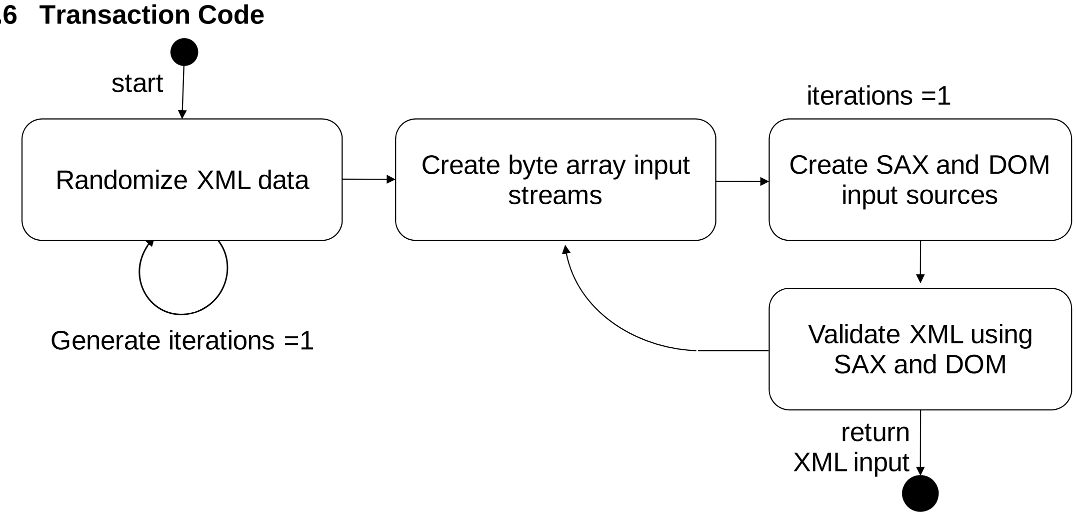
10.6.4 Required Initialization

A initialization time, both XML and XML schemata files are read in from disk and saved in a buffer for future use. (There will be no further disk IO once this is completed.) A randomization algorithm is applied to the original XML data on the fly before for each transaction execution to create variations in parsing without modifying file size or complexity.

10.6.5 Configuration Parameters

`xml-schema-dir`	Specifies the directory of the XML schema file.
`xml-schema-file`	Specifies the name of the XML schema file.
`xml-dir`	Specifies the directory of the XML file.
`xml-file`	Specifies the name of the XML file.
`enable-idc`	Enables/disables memory scaling using input data caching (IDC). Must be set to false.
`iterations`	Number of executions per transaction.
`debug-level`	Value governs the volume of debug messages printed during execution.
`input-generate-iterations`	Number of XML file randomization iterations.

10.6.6 Transaction Code



10.7 Memory Worklet: Flood

10.7.1 General Description

The Flood workload is based upon STREAM, a popular benchmark that measures memory bandwidth across four common and important array operations. For the *long* (64-bit) integer arrays used in Flood, the following amounts of memory are involved per assignment:

1. **COPY:** *a(i) = b(i)*
   -- 8 bytes read + 8 bytes write per assignment = 16 bytes / assignment
2. **SCALE:** *a(i) = k * b(i)*
   -- 8 bytes read + 8 bytes write per assignment = 16 bytes / assignment
3. **ADD:** *a(i) = b(i) + c(i)*
   -- 16 bytes read + 8 bytes write per assignment = 24 bytes / assignment
4. **TRIAD:** *a(i) = b(i) + k * c(i)*
   -- 16 bytes read + 8 bytes write per assignment = 24 bytes / assignment

The Flood score is based upon the aggregate system memory bandwidth calculated from the average of these four tests multiplied by the amount of physical memory installed in the SUT. While Flood is based upon STREAM, it uses no STREAM code and is implemented wholly in Java.

Flood enhances STREAM in a variety of important ways.

1. Flood rewards systems with large memory configurations by scaling results based upon physical memory size.
2. Flood is designed to fully exploit the memory bandwidth capabilities of modern multi-core servers. Flood is multithreaded and threads are scheduled to operate concurrently during bandwidth measurements ensuring maximum throughput and minimizing result variability.
3. Flood requires little to no user configuration, yet automatically expands the data set under test to fully utilize available memory.

Measuring aggregate system memory bandwidth on large servers with many cores and multiple memory controllers is challenging. In particular, run-to-run variability is often unmanageable with existing memory bandwidth benchmarks. Flood minimizes run-to-run variation by taking three memory bandwidth tests back-to-back and discarding the first and last tests. This ensures that all threads are running under fully concurrent conditions during the middle measurement which is used in Flood scoring calculations.

Flood scores scales linearly with a SUT's aggregate memory bandwidth as well as with the SUT's physical memory configuration. CPU, storage and network performance have little to no impact on Flood scores.

Since the Flood workload always deploys a fixed number of iterations and the amount of memory under test will automatically adjust to fully utilize installed DRAM, run time will vary depending upon system configuration. On a 2.2GHz, 24-core SUT with 24 threads and 48GB of physical memory, Flood takes about 20 minutes to complete. Run time varies proportionally with the amount of physical memory installed in the SUT. Run time is also impacted by the overall thread count.

10.7.2 Sequence Execution Methods

FixedIterationsDirectorSequence – Flood is executed for a given set of iterations specified within *config.xml.*

10.7.3 Metric

Score = aggregate system memory bandwidth (GB/s) * physical memory size (GB)

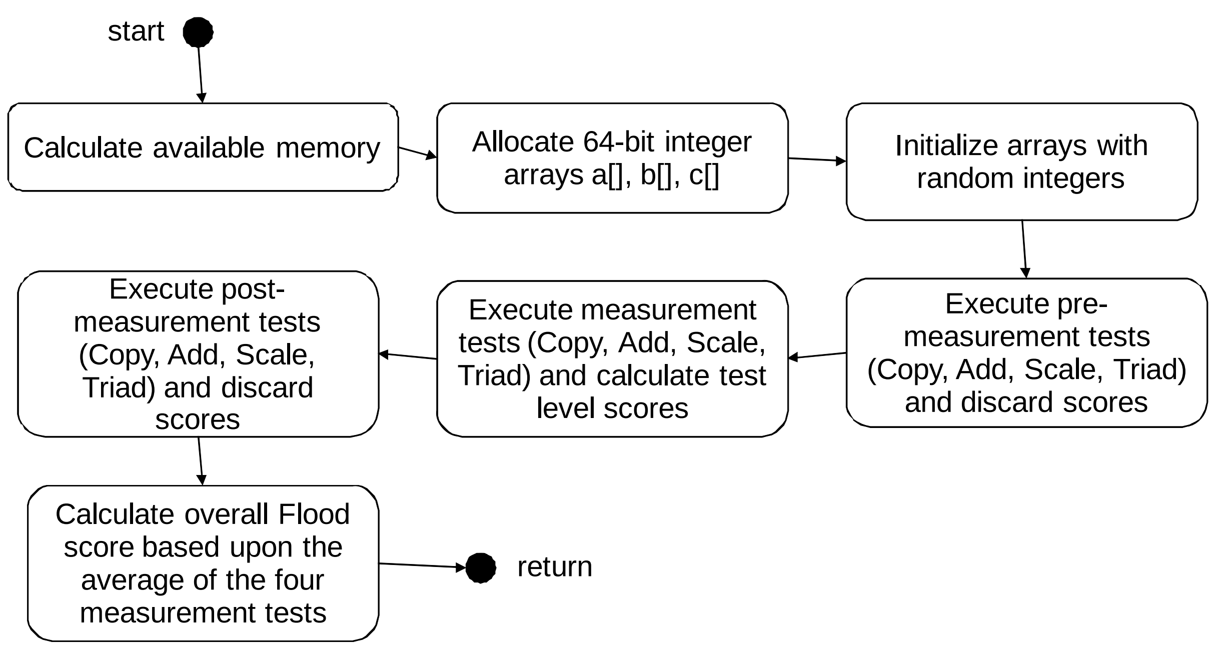
10.7.4 Required Initialization

Flood calculates the amount of memory available to the thread and creates three 64-bit (*long*) integer arrays, a[], b[] and c[], to completely utilize all available space. These arrays are initialized with random data. To ensure full load concurrency during bandwidth measurements, a complete set of pre-measurement tests is launched prior to an identical measurement period followed by identical post-measurement tests. Only the test results for the measurement period are utilized for Flood score generation.

10.7.5 Configuration Parameters

`memory-under-test`	The default value of "-1 MB" turns on automatic configuration of the data set size. However, the user can override this behavior and explicitly define the amount of memory to test per JVM. Valid values are (san quotation marks): "200 MB", "1.1 GB", "10000000 B".
`iterations`	Flood internally iterates the number of memory bandwidth tests based upon the value of the iterations parameter. The default is 100.
`debug-level`	Detailed diagnostic information can be enable through the *debug* parameter. Valid values are 0 = no additional debug information (default), 1 = debug information turned on, 2 = detailed debug information.
`return-bandwith`	The raw, aggregate system memory bandwidth calculated by Flood can be obtained by setting the parameter return-bandwidth to "true" in which case Flood will return measured memory bandwidth instead of a score. The default value is "false".

10.7.6 Transaction Code



10.8 Memory Workload: XmlValidate

10.8.1 General Description

The XML validate workload implements a transaction that exercises Java's XML validation package javax.xml.validation. Using both SAX and DOM APIs, an XML file (.xml) is validated against an XML schemata file (.xsd). To randomize input data, an algorithm is applied that swaps the position of commented regions within the XML input data.
Memory scaling in XmlValidate is done through a scheme known as input data caching (IDC). In IDC, the universe of possible input data (here, randomized XML file data) is pre-computed and then cached within memory before the start of the workload. During workload execution, the input data for a particular transaction instance is then chosen randomly and retrieved from this cache rather than computed on the fly.

10.8.2 Sequence Execution Methods

Graduated Measurement Sequence

10.8.3 Metric

Transactions Per Second * Cache size * Cache size scaling factor

10.8.4 Required Initialization

A initialization time, both XML and XML schemata files are read in from disk and saved in a buffer for future use. (There will be no further disk IO once this is completed.) IDC initialization follows during which all possible input data sets are pre-computed and cached in memory. For each input data set, a randomization algorithm is applied to the original XML data to create variations in parsing without modifying file size or complexity.

10.8.5 Configuration Parameters

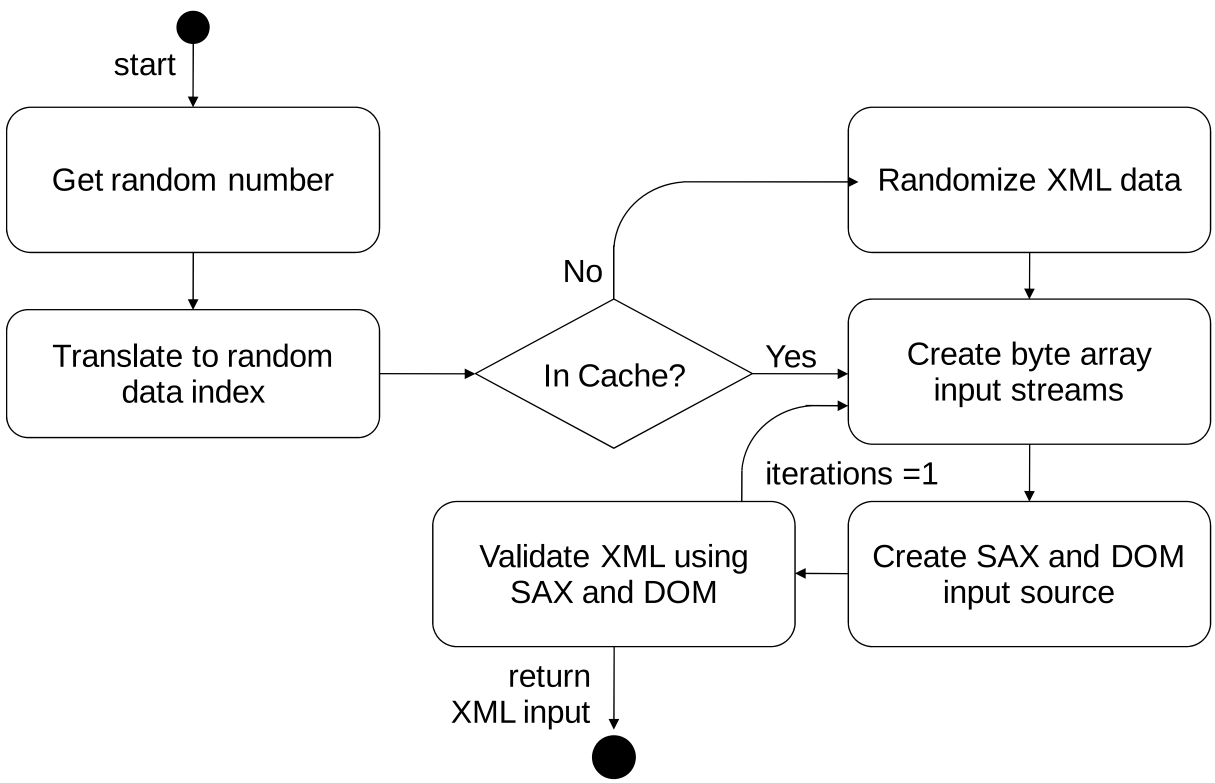
XmlValidate parameters:

`xml-schema-dir`	Specifies the directory of the XML schema file.
`xml-schema-file`	Specifies the name of the XML schema file.
`xml-dir`	Specifies the directory of the XML file.
`xml-file`	Specifies the name of the XML file.
`enable-idc`	Enables/disables memory scaling using input data caching (IDC). Must be set to false.
`iterations`	Number of executions per transaction.
`debug-level`	Value governs the volume of debug messages printed during execution.
`input-generate-iterations`	Number of XML file randomization iterations.

Additional IDC configuration parameters:

`store-type`	Specifies the algorithm to use in generating data when a cache miss occurs.
`locality-distribution`	Specifies the probability distribution to use when randomly choosing input data indices.
`data-store-size`	Specifies the size of the universe of possible input data.
`data-cache-size`	Specifies the size of the input data cache.
`data-cache-report-interval`	Governs the frequency of output messages on cache hit/miss ratio.
`custom-score-policy`	Specifies the algorithm to use in computing custom score reflecting cache size configuration.
`data-cache-size-scale factor`	Specifies the scaling factor to use in the DataCacheSizeMultiplierGB custom scoring algorithm.
`data-cache-to-heap-ratio`	Ratio of cache size to JVM heap size used in automatic cache sizing.

10.8.6 Transaction Code



10.9 Storage IO Workload

10.9.1 General Description

The Storage-Workload has four different transactions, two random and two sequential transaction-pairs. Each pair has a write and a read transaction.

10.9.2 Sequence Execution Methods

[Graduated Measurement Sequence] or [Fixed Iteration Measurement Sequence]

10.9.3 Metric

Score name and definition of what the score value represent

10.9.4 Required Initialization

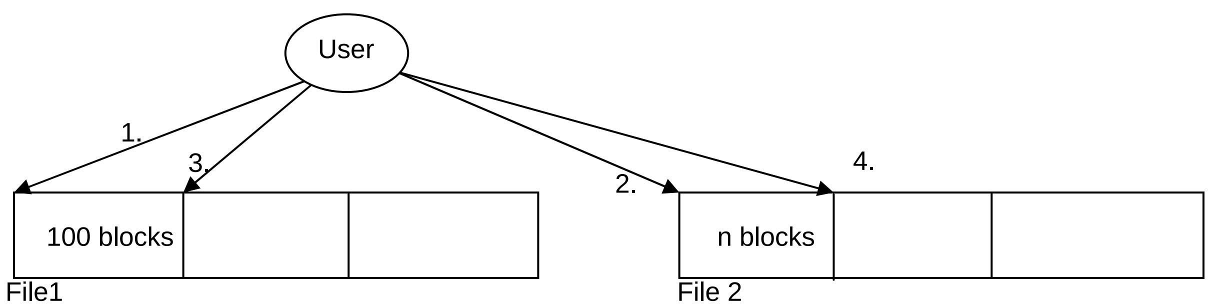
A set of files is created before execution of the transaction

10.9.5 Configuration Parameters

`file-size-bytes`	size of a file.
`file-per-user`	number of files opened by each user.
`file-path`	location of the files - In this example the path is “D:\data\”, please note that the files always reside in a subfolder called “data”.
`max-count`	amount of blocks that are accessed by the sequential transaction in one file before the next file is addressed.

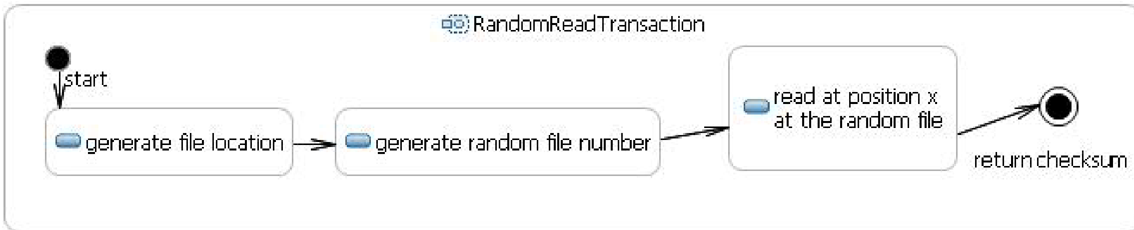
Example:

```
<file-size-bytes>1000000<file-size-bytes>
<file-path>D:\</file-path>
<file-per-user>2</file-per-user>
<max-count>100</max-count>
```

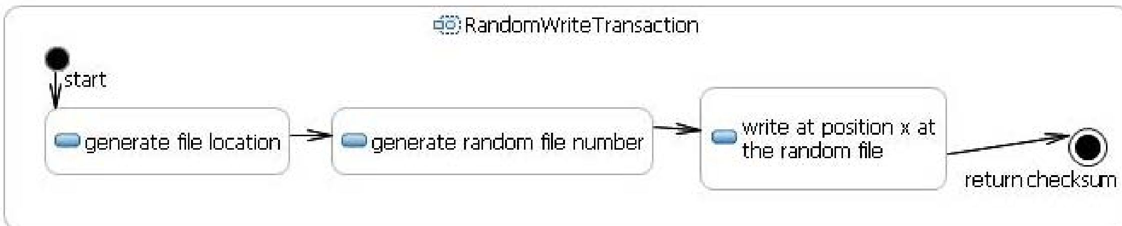


[Figure 7: File Example (2 files per user and max-count of 100)]

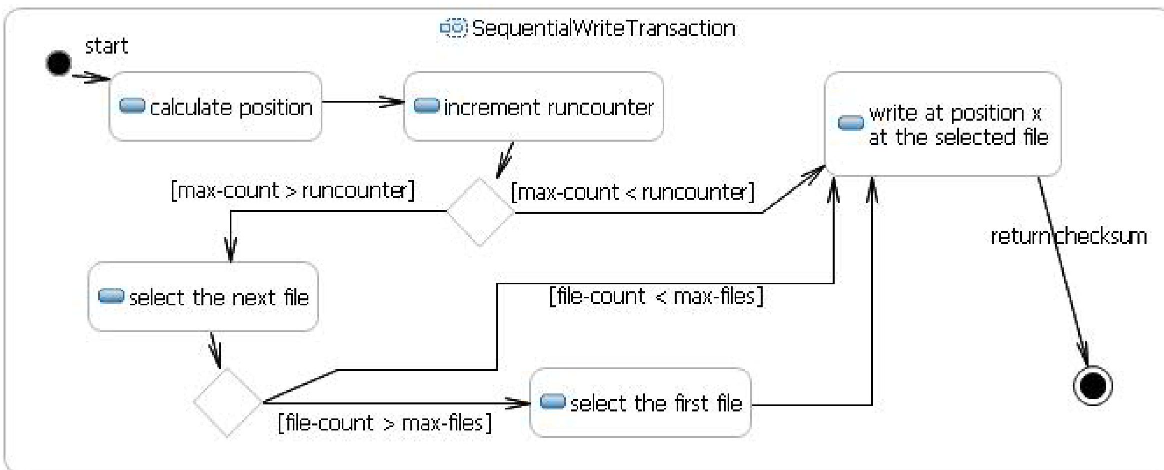
10.9.6 Transaction – Code 1 - RandomRead



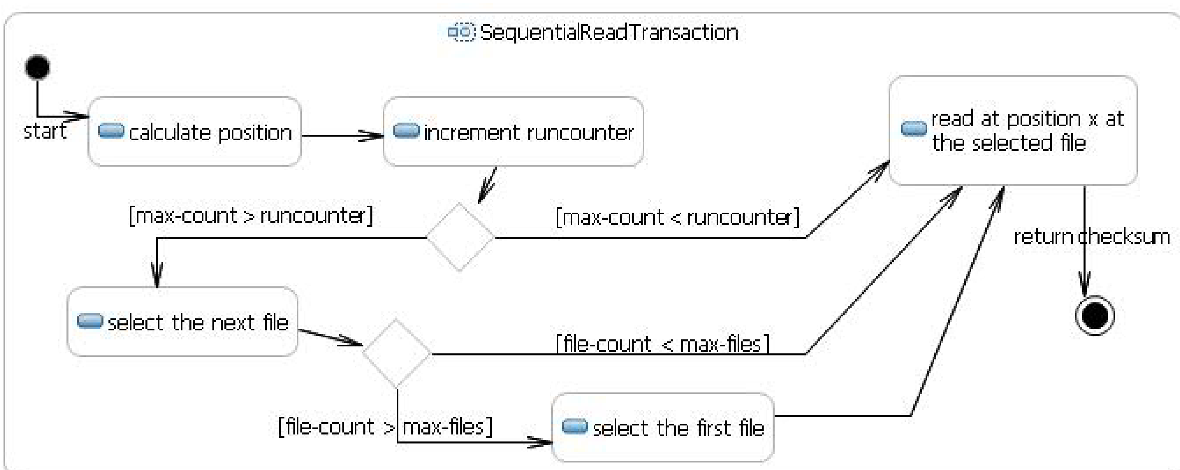
10.9.7 Transaction – Code 1 - RandomWrite



10.9.8 Transaction – Code 2 – SequentialRead



10.9.9 Transaction – Code 2 – SequentialWrite



10.10 System Worklet: CSSJ

10.10.1 General Description

CSSJ is an Online Transaction Processing (OLTP) workload, and represents a Server Side Java application. It is based on the SSJ workload in SPECpower_ssj2008, which was based on SPECjbb2005, which was inspired by the TPC-C specification; however, there are several differences between all of these workloads, and CSSJ results are not comparable to any of these other benchmarks.

The System Worklet exercises the CPU(s), caches, and memory of the UUT. The peak throughput level is determined by maximum number of transaction of the above type the system can perform per second. Once the peak value of the transactions is determined on a given system, the worklet is run from peak (100%) down to the system idle in a graduated manner.

The performance of the System Worklet depends on the combination of the processor type, number of processors, their operating speed, and the latency and bandwidth of the memory subsystem of the system.

CSSJ includes 6 transactions, with the approximate frequency shown below:

- New Order (30.3%) – a new order is inserted into the system
- Payment (30.3%) – record a customer payment
- Order Status (3.0%) – request the status of an existing order
- Delivery (3.0%) – process orders for delivery
- Stock Level (3.0%) – find recently ordered items with low stock levels
- Customer Report (30.3%) – create a report of recent activity for a customer

10.10.2 Sequence Execution Methods

Graduated Measurement Sequence

10.10.3 Metric

Transactions per second

10.10.4 Required Initialization

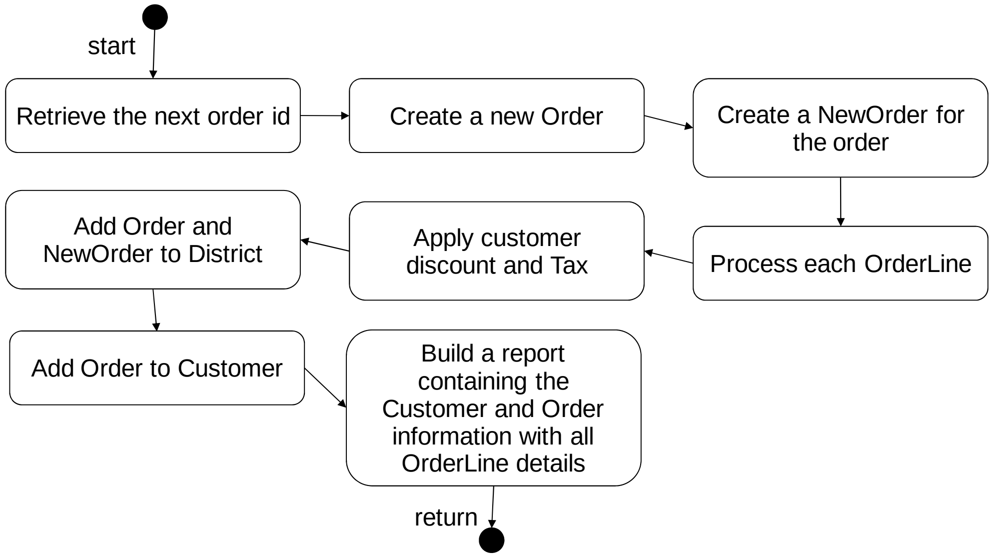
Each user represents a warehouse. During initialization, each warehouse is populated with a base set of data, including customers, initial orders, and order history.

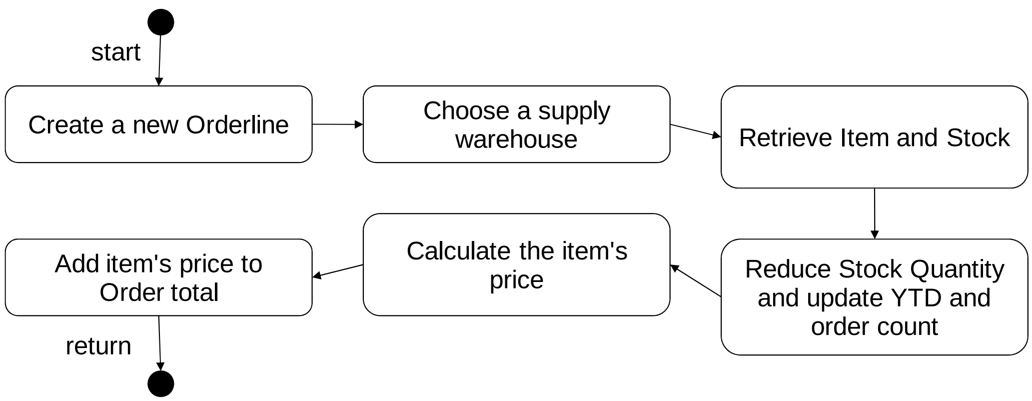
10.10.5 Configuration Parameters

The CSSJ workload does not have any supported configuration parameters.

10.10.6 New Order Transaction

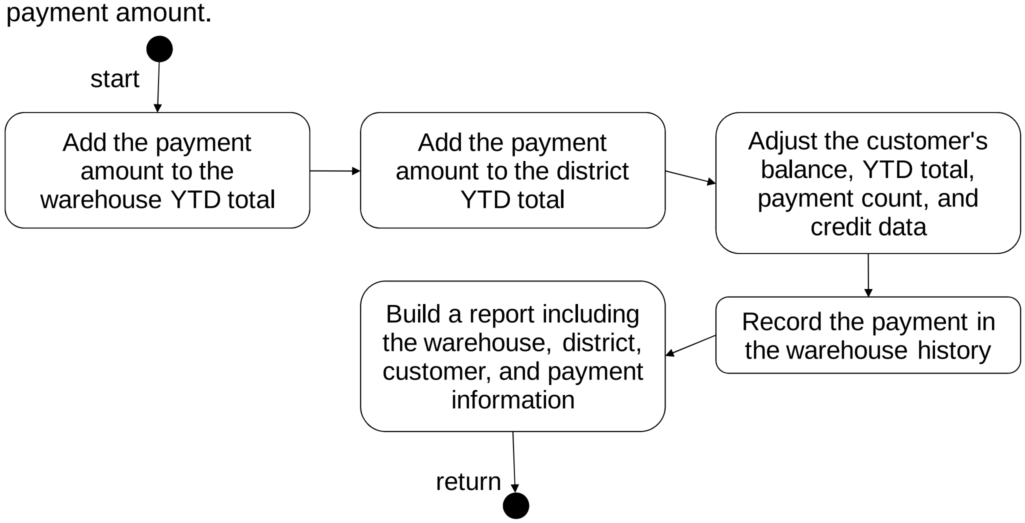
The input for a New Order Transaction consists of a random district and customer id in the user's warehouse, and a random number of orderlines between 5 and 15.





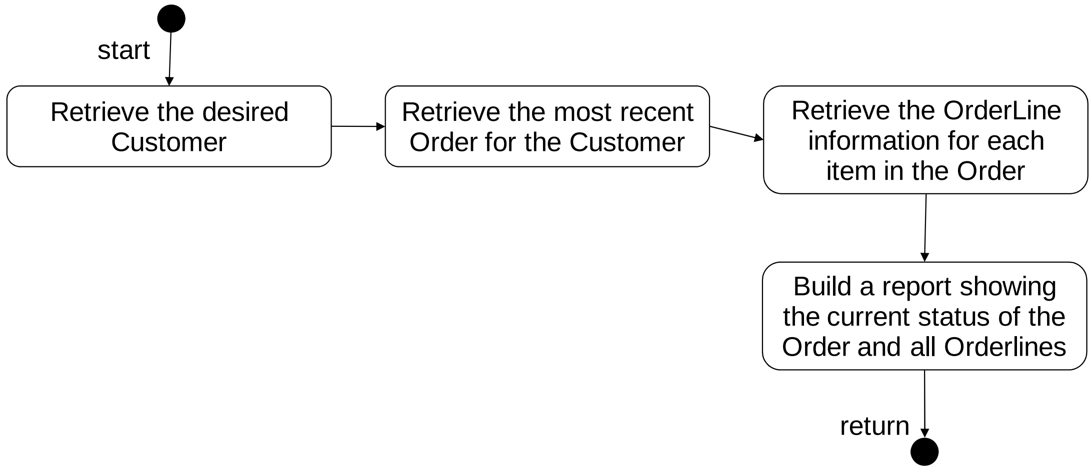
10.10.7 Payment Transaction

The input for a Payment Transaction consists of a random district from the user's warehouse, a random customer id or last name (from either the user's warehouse or a remote warehouse) and a random payment amount.



10.10.8 Order Status Transaction

The input for an Order Status Transaction consists of a random district and either a customer id or last name from the user's warehouse.



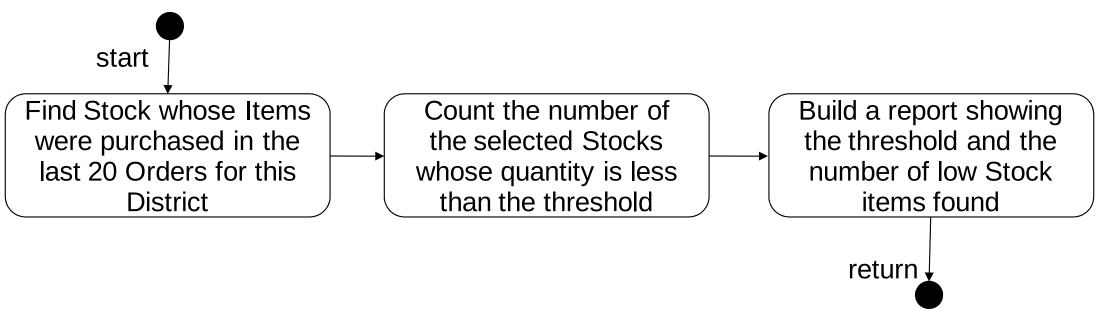
10.10.9 Delivery Transaction

The input for a Delivery transaction is a random carrier id.

[The activity diagram is work in progress]

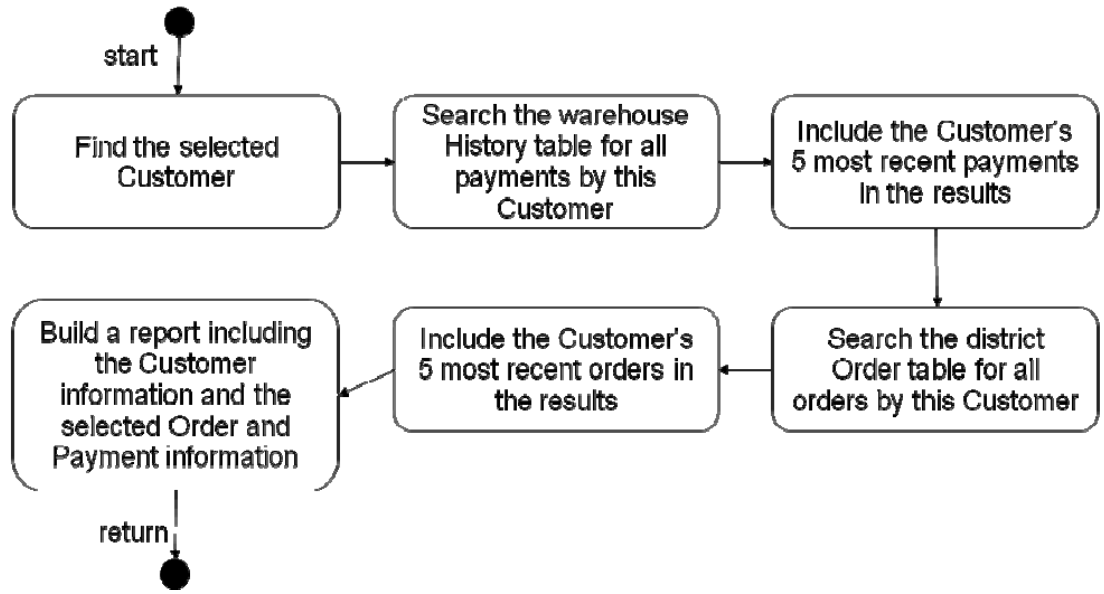
10.10.10 Stock Level Transaction

The input for a Stock level transaction is a random district from the user's warehouse and a random “low level” threshold between 10 and 20.



10.10.11 Customer Report Transaction

The input for a Customer Report transaction consists of a random district from the user's warehouse and a random customer id or last name (from either the user's warehouse or a remote warehouse).


start
Find the selected Customer
Search the warehouse History table for all payments by this Customer
Include the Customer's 5 most recent payments in the results
Search the district Order table for all orders by this Customer
Include the Customer's 5 most recent orders in the results
Build a report including the Customer information and the selected Order and Payment information
return

