Parasoft Load Test Introduction

In this section:

About Load Test

Parasoft Load Test allows you to reuse your SOAtest functional tests to verify application performance and functionality under heavy load. Parasoft Load Test features:

- **Centrally-managed load test configuration/execution** with seamless integration into Parasoft SOAtest. This is aligned with how teams and roles are typically structured within an organization.
- The ability to **load test complete end-to-end test scenarios**—from the web interface, through services, to the database. Every protocol and test type available in Parasoft SOAtest is supported in Parasoft Load Test.
- Support for **load testing non-Parasoft components such as JUnits** or lightweight socket-based components. This provides an integrated solution for your various load testing needs.

The solution not only monitors the server's response rate with the specified number and mixture of simultaneous requests, but also verifies whether functionality problems occur under load. Prebuilt scenarios can be used to verify robustness and scalability. You can easily customize these scenarios to use different test cases, load levels, load distributions, and so on. You can also distribute virtual users across remote server machines to simulate extreme loads and/or test from different locations. Support is also provided for load testing non-Parasoft components such as JUnits or lightweight socket-based components. This provides teams an integrated solution for their various load testing needs.

To help you collect network information and system performance data during load testing, Parasoft's solution provides built-in and extensible monitoring capabilities. For instance, supported monitors include perfmon, SNMP, rstat, WebSphere, WebLogic, JBoss, Tomcat, and remote monitors. The solution's extensible and deployable monitoring framework allows you to extract just about any set of metrics from the system that is being subjected to load, graph and correlate those metrics visually to identify causes of performance issues, and apply expected quality of service policies on those metrics to get a high-level view of the system's compliance with performance and reliability requirements.

In addition, Parasoft Load Test provides built-in support for testing JUnits as well as a framework for load testing any component that implements the Parasoft load test component API; for example, it can allow load testing with lightweight Socket-based components that implement the Parasoft component API. This allows the load test to be specialized and tailored for the various unique complexities that organizations face in performing performance validation.

System Requirements

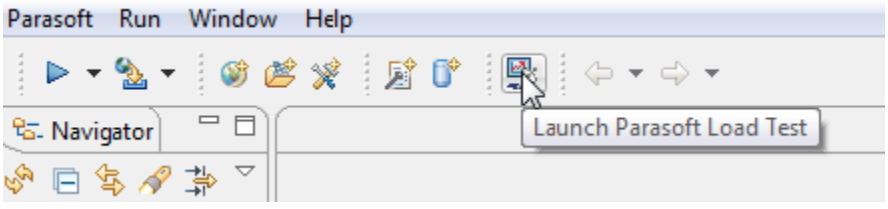
Load Test requires 4GB for a single Load Test process and 2GB for each additional Load Test process. 4GB of system memory for each Load Test process is recommended.

Installation

The Parasoft SOAtest installer installs both Parasoft SOAtest and Parasoft Load Test.

Starting Load Test

You can open Load Test by clicking the **Launch Parasoft Load Test** toolbar button.



Starting Load Test from the Command Line

You can pass system properties to Load Test when starting it from the command line.

Dependency Synchronization Settings

The following setting causes Load Test to ignore external dependencies. The dependencies will not show in the Machine Dependencies view of the machine configuration panel.

```
lt -J-Ddependencies.ignore=true
```

The following setting causes Load Test to show the dependencies but disables them by default.

```
lt -J-Ddependencies.default.unselect=true
```

Asking for the Java Version

Starting Load Test with the `-ask` flag will prompt you to specify the Java executable you want to run. This is helpful if you need to run with a different version or alternate distribution of Java. Connecting to a WebSphere application over SSL, for example, requires the IBM JRE. You can download the IBM JRE and use the `-ask` option when starting Load Test, which will prompt you to specify the alternate Java. See IBM WebSphere MQ for additional information.

Windows:

```
loadtest -ask
```

Linux and MacOS:

Load Test searches for a loadtest.config configuration file whenever it is started. If the file does not exist, Load Test and will create the file and print the location to the console. You will need to manually edit the configuration file if you want to change the Java runtime. For example:

```
atrujillo$ ./loadtest
Found a new Parasoft Load Test installation. Parasoft Load Test will be configured.
Using /Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/jre/bin/java
Parasoft Load Test setup in /Users/atrujillo/Downloads/parasoft/soatest/9.10/Parasoft Load Test.app/Contents
/MacOS/../../../Parasoft SOAtest.app/Contents/ParasoftSOAtest/eclipse/plugins/com.parasoft.ptest.libs.web_9.
10.0.20161130/root completed successfully
Starting Parasoft Load Test...
```

The console may not print the location of the configuration file after the first start up, but it is created in the following directory so you can edit it any time: < SOAVIRT_INSTALL>/plugins/com.parasoft.ptest.libs.web_<VERSION>/root/

Load Testing SOAtest Functional Tests

Parasoft Load Test allows you to run your functional tests under increased loads. Parasoft Load Test repeatedly executes selected test suites with the specified number of virtual users or number of hits per second (hit rate).

> **Web Load Testing**
>
> If you want to use your browser-based functional tests for browser-less web load testing, use SOAtest to configure them for this application.For details, see Preparing Web Functional Tests for Load Testing.

The goal of load testing is to verify application/service performance and functionality under heavy load. It allows you the ability to run test suites in parallel with multiple threads and multiple users.

The best way to start load testing is to have multiple test clients run the complete functional test, including request submissions and response verifications. When load testing ignores the functionality verification process and focuses solely on load-rate metrics, it risks overlooking critical flaws such as functionality problems that surface only under certain loads.

The best way to start load testing is to have multiple test clients run the complete functional test, including request submissions and response verifications. When load testing ignores the functionality verification process and focuses solely on load-rate metrics, it risks overlooking critical flaws such as functionality problems that surface only under certain loads.

To thoroughly test performance, a functional test suite should be run under a variety of different scenarios to check how the different types of loads are handled. For example, the test could check functionality and response time under different degrees of load increases (sudden surges versus gradual ramp-ups) or different combinations of valid and invalid requests. If the load tests reveal unacceptable performance or functionality under load, the next step is to diagnose and repair the source of the bottleneck. Sometimes, the problem is caused by a fundamental algorithmic problem in the application, and the repair could require something as painful as an application redesign and rewrite. Other times, it is caused by some part of the infrastructure (the Web server, the SOAP library, the database, and so forth). In these cases, fixing the problem might be as simple as changing a configuration or as complex as changing the architecture.

Because fixing performance problems sometimes demands significant application or system changes, it is best to start load testing as soon as possible. By starting early, you can diagnose and fix any fundamental problems before it is too late to do so without a major rewriting or rebuilding nightmare.

Test Your Own Applications--Not Public Applications!

Parasoft does not recommend or condone performing load tests on public applications (including the applications that are referenced in the SOAtest tutorial). Please use it to load test only your own applications, or the parabank sample application used in the Load Test tutorial.

Migrating Tests from Earlier Generations of Parasoft SOAtest or WebKing

If you have load tests configured in SOAtest 5.5 or earlier, they can be imported and are fully supported. Load tests from Parasoft WebKing are also supported-- they should be opened in Parasoft SOAtest, then configured and validated for load testing as described in the SOAtest user's guide.

Configuring Dependencies on External JAR files or Class Folders

If you will be load testing a SOAtest project that uses Extension Tools with dependencies on external JAR files or class folders, do the following to ensure that these SOAtest Extension Tools find their dependencies in Load Test:

1. Choose **Parasoft> Preferences**.
2. Under **System Properties**, add the jars or class folders to the System Properties classpath entries (if they are not already there).
3. Close SOAtest.
4. Close Load Test if open.
5. Open Load Test. The jars from SOAtest system preferences will be imported into Load Test—even though they are not shown in the Load Test preferences.