Using WebLOAD to Monitor Your Production Environment

Your pre-launch performance test scripts can be reused for post-launch monitoring to verify application performance. This reuse can save time, money and resources.

Executive Summary

Your company invests in creating a robust set of performance test scripts to ensure the quality of your web applications before they go live. Then when the application is deployed you need to implement a performance monitoring solution. Radview WebLOAD is designed to fulfill both of these functions—pre-launch performance testing and post-launch monitoring. This approach facilitates reuse of knowledge, skills, manpower and tools. It also assures quality by transferring the lessons learned in testing to the monitoring environment.

In this white paper we will review the application monitoring process and the monitoring features of WebLOAD. We will also discuss considerations when configuring your monitoring tool, creating monitoring scripts and handling notifications.
Overview

What is APM?
Application Performance Management (APM) can be defined as the process of detecting, diagnosing, reporting and remedying the health of an application’s performance. Application performance relates to the reliability and the speed with which data is delivered to the end user and with which transactions are completed by the application via a particular network, application and/or web services infrastructure.

The two main methods for assessing application performance are:

- Measuring resource usage at the server side
- Measuring response time of the application itself from the client side

The first method, referred to as Systems Management or Systems Monitoring, has been in use for as long as computers have been used for business applications and still constitutes an important aspect of APM. This method also involves measuring various performance counters from the hardware (for example, network devices), the operating system and infrastructure software components like database systems and web servers.

The second method, measuring the application’s behavior from the user’s perspective is considered the most valid method of assessing the performance of an application in production. This method is used whenever it is both possible and financially acceptable to do so.

APM Solutions
APM solutions require a large investment in setting up the environment, creating monitoring scripts within the tool and purchasing the monitoring tool itself. In this document we present a different approach to APM: reusing the pre-launch test scripts for post-launch monitoring of the application in production. We will outline the benefits of this approach and the methodology for using WebLOAD to monitor your application in the production environment.

The benefits of using the same tool for both pre-launch testing and post-launch monitoring include:

- Reuse of knowledge, skills and lessons learned
- Reuse of technologies and test scripts
- Reduced expenditure on tools and manpower
The Monitoring Process: Overview
While some application monitoring is reactive and only allows you to deal with problems when they are reported by users, most IT managers now require a proactive approach that identifies problems as early as possible.

Furthermore, traditional monitoring focuses on system infrastructure such as servers, network devices, operating systems and databases. Although this provides critical information about the availability and health of your system, it does not provide monitoring of the application at the level of the end user. This is problematic because it is possible for your system to be healthy and for problems to still exist at the application level resulting in an unacceptable user experience. Basing your monitoring setup on WebLOAD ensures that the end-user experience is tracked and not only the system infrastructure.

Monitoring Web Applications with WebLOAD
The sections below summarize some of the key functions supported by WebLOAD for Application Performance Management.

Confirming Availability
Check for the availability of a service, an application, or a page. WebLOAD can be set up to check availability once every defined interval and report if the application is non-responsive. By simply repeating the recorded business process, WebLOAD will request every page and resource and will report if an item is absent.

Detecting Performance Degradation Before It Hits the Red Line
Sometimes an application is available—and able to pass a simple availability test—but is actually running more slowly than required. This could be a case of gradual performance degradation. WebLOAD can check the performance or responsiveness of a service, an application or a specific page over time and provide early warning of performance degradation. You can set thresholds according to the Service Level Agreement (SLA) defined by your management and business users. You can then test whether each resource performs within the SLA. You can also test the response times of specific resources and/or the overall average response time of your application.

In order to detect gradual performance degradations, reporting for these cases should not be binary. Rather, we recommend defining at least three ranges for each threshold:

- Normal (green): when response times are OK
- Suspect (yellow): when response times are still within the desired SLA, but approaching maximum legal values (like the Fuel Empty light in your car – it doesn’t really mean you have no fuel left, but rather that you should get to the nearest gas station)
• Problem (or red): when response times exceed the defined SLA.

Verifying Application Behavior
The process of preparing your test scripts includes modeling the application by recording user scenarios and defining the expected output. Sometimes a service will appear to be available, even when under high load, but will in fact behave incorrectly. To identify these cases your test script should check the data returned by the application and verify that the contents are valid and complete.

Returned data can be HTML pages (in today’s applications most pages are generated dynamically based on the results of the last user action), web service results, or any other information returned by your application. Your test script can use string manipulation tools to parse the data returned and check for expressions that indicate success or failure.

A more efficient verification can be implemented using the DOM API provided with WebLOAD. This API supports easy programmatic access to elements within HTML and XML documents. With this approach the script directly accesses a clearly defined data structure, rather than searching within a string, and is thus more efficient.

Monitoring System Infrastructure Performance
In addition to application-level monitoring, WebLOAD also supports monitoring of traditional system parameters, such as CPU utilization, memory consumption, disk space, etc., from servers, network hubs and infrastructure software like operating systems, databases and web servers. Although monitoring of these system components is usually covered by other, more traditional system management tools, it can be wise to use a single APM tool for monitoring both system and application performance. This approach means that for each monitoring session, you have a single results database with both application and system performance measurements aligned to the same time line. This makes it much easier to isolate the source of performance problems. WebLOAD supports monitoring of system and server-side parameters by using its PMM (Performance Measurements Manager) module.

Reporting Problems
Reports should be generated both when acute problems arise and on an ongoing basis. These periodical reports are useful in troubleshooting issues that evolve over time. It is often difficult to pinpoint the source of these issues; periodic reports provide a summary of transaction statistics that you can compare to a baseline. This can make all the difference in identifying and resolving slow performance degradation.

WebLOAD includes an advanced reporting and analysis module, WebLOAD Analytics, that can be used for this purpose. Use it to generate reports from the results of your daily monitoring session, or—even better—use it to generate regression reports that compare the results of several sessions in a single report.
Considerations in Configuring Your Application Monitoring

Overview
While pre-launch testing and post-launch monitoring have many common goals and practices, there are a number of differences which should be reflected in the configuration of your load testing tool for application performance monitoring. The section below reviews these issues.

Multiple Virtual Clients vs. Multiple Topology Configurations
While load testing requires simulation of a large number of virtual clients, this is not recommended for performance monitoring. The introduction of large numbers of synthetic users to a production system not only wastes resources but can also skew the results of the monitoring. The monitoring process should be as non-intrusive as possible. Thus it is recommended to run monitoring scripts with only one Virtual Client.

On the other hand, the virtual clients configured for performance monitoring should represent the various Internet topologies used for access to your application by real life users. Internet topology issues include geographical location, ISP, connection type and connection speed. To support this, load generator servers can be located at multiple sites.

Internal/External
When creating your monitoring setup you’ll need to choose to monitor inside the firewall, outside the firewall or both. If you need to monitor links to external sites or to get information about network latency for remote users, we recommend including virtual users at these locations as part of your setup. If it’s only your application’s performance that you’re after, and the network performance is not important, then monitoring inside the firewall should be enough.

Scheduling
Application performance monitoring should take place whenever the application is in use. Choose either a 24x7 approach or a smaller monitoring window if that reflects the actual application usage. For example, if your application is deployed via your organization’s intranet and mainly used during work hours, then it might be sufficient to monitor it during these times. On the other hand, if your application is deployed on the Internet and serves users around the world, then it would be wise to monitor it constantly.

It is important to note that, by their nature, load testing tools log results to a database once per session. To ensure data integrity we recommend initiating a new load session every 24 hours so that you get separate result databases each day.

When you have determined the optimal schedule for your APM setup you can use WebLOAD’s extensive command line interface (CLI) with an external scheduling engine (like Windows
Scheduled Tasks system tool or any other 3rd party scheduler). Simply prepare your WebLOAD test scripts and templates and provide them as parameters to WebLOAD via a shell script file.

After running the scripts you can also use the WebLOAD Analytics CLI to generate reports.

Selecting Modules to be Monitored
For the sake of modularity and maintainability we advise separating each business process (or probe) into a separate script. In this way you will be able to control which probes you want active and which not. Let’s say an application or part of an application is down for maintenance, you simply exclude the corresponding probe from that day’s run.

Resolution
Carefully consider the best time resolution for your probes. Should you check all probes and performance counters every second, every 10 seconds or maybe once a minute is enough? This will affect the size of the result set collected in the tool’s database. If you go for a shorter interval, keep in mind that you may receive multiple notifications for a single error condition.

Configuring Notifications
It is important to control your notifications so that your staff is not bombarded with numerous alerts. You should decide in advance who to notify about which problems and which notification methods to use. To establish an effective notification policy keep in mind that any monitoring setup may generate false positives.

Defining Tolerance Levels
Usually, performance testing tools are configured to stop running a test when there is a critical application failure. However, APM is a continuous process that should be much more tolerant to application errors and should report them but not stop running. This can be configured in WebLOAD using the pass/fail definition option.

Scripting Considerations
Although the functions required for load testing and for application performance monitoring are similar, the environmental factors are different. This should be reflected in scripting practices. The following sections review issues that should be taken into consideration when scripting for monitoring.

The Importance of Reliable Scripts
Monitoring scripts may need to run for days without re-starting and therefore require a higher-level of quality than load-testing scripts. A buggy script could lead to an inefficient workflow. Implement high quality standards for your monitoring scripts.
Selecting Modules to be Monitored
For the sake of modularity and maintainability we advise separating each business process (or probe) into a separate script. In this way you can control which probes are active and which are not. For example, if an application or part of an application is down for maintenance, you simply exclude the corresponding probe from that day’s run.

Setting up Transactions
Every application has more critical and less critical elements. We recommend defining your application’s most critical business processes as transactions. By defining transactions you also get automatic collection of performance counters for this part of the application. These counters include timers and TPS (Transaction Per Second) statistics which can be compared to predefined SLAs and trigger alerts when necessary. Clearly detailed statistics are also of great value in post-facto analysis of the source of a problem.

Generic Functions
Write common code elements as generic functions and include them in the actual business process scripts. For example, it is a good idea to create a JavaScript module for generating email notifications and including it in all relevant scripts.

Varying Input Parameters
To achieve effective test coverage that can really expose problems, we recommend using varying input parameters within your scripts. For example, rather than testing an application’s login function repeatedly with the same user name and password, use several different user names and passwords. These can be read by the script from an external file and used to parameterize the login transaction. It is important to find the right balance between too little and too much variety, as too much variety can make it difficult to troubleshoot issues that arise over longer periods of time.

WebLOAD Support for Notifications
Someone needs to know when a problem arises with your application and to deal with it. So application monitoring is not complete without real-time notifications. The sections below describe the notification mechanisms available with WebLOAD.

Notification Manager
WebLOAD’s Notification Manager allows you to define rules for sending notification emails when certain measurements meet a predefined condition. You can select from a list of measurements and indicate that a notification should be sent if their value is equal to, less than, or greater than a specified value. If the condition is met, WebLOAD sends an email to an address defined in the notification rule.
Email Support
WebLOAD’s Notification Manager is designed to allow quick setup of notification rules that result in static email messages. To create more complex notification rules that generate email messages with dynamic content (for example, including data provided with an error condition) you can use the SMTP protocol Building Block included as part of WebLOAD’s Internet Productivity Pack (IPP).

Open Interfaces
In addition to email, WebLOAD supports any other form of notification using its open interfaces to call any third party Java or ActiveX component. For example, you can easily integrate any existing component to send notifications via SMS, create entries in another monitoring tool’s queue, or any other method available. More information about WebLOAD’s open interfaces can be found in the WebLOAD SDK documentation.

Logging
We recommend keeping the data sent with notification concise and to provide additional information only if vital. You can fetch expanded information about an issue from the WebLOAD database after the load session is completed (if you import the load session into WebLOAD Analytics). You can search for the record required using the event’s timestamp as a key.

If you need this extra information to be available while the load session is still running, you can write your own logs. WebLOAD supports writing to external text files and also to database systems (Oracle, MSSQL, MySQL) via an existing interface that is provided with the tool. See the Database group in the WebLOAD IDE Toolbox for a set of Building Blocks that allow you to easily communicate with an external database.

Writing to an external file can also be useful if you encounter an application-level error in one of your application’s pages. You can save the entire HTML document into an external text file for later inspection.

Summary
As many companies using WebLOAD have already discovered, it is easy to take a step beyond pre-launch load testing toward post-launch application performance monitoring. Reusing the same tool for both functions assures efficient use of your resources.

Your WebLOAD scripts can easily be modified for use as part of an APM solution. The tool’s strong analysis and reporting functions assure high quality monitoring and trouble-shooting capabilities. Following the configuration and scripting guidelines outlined in this document will help assure effective Application Performance Management.
RadView Contact Information:

North America
RadView Software, Inc.
991 Highway 22 West, Suite 200
Bridgewater, NJ 08807
Phone: 908-526-7756 Fax: 908 864 8099
Toll Free: 1-888-RADVIEW

International:
RadView Software LTD
14 Hamelacha St, Rosh Haayin 48091, Israel
Phone: +972-3-915-7060 Fax: +972-3-915-7683

www.radview.com