<00_Intro:>

As a QA Professional, upon the discovery of a performance problem, which of the following statements would you like to make to your developers:

Our application exhibits a slowdown during the checkout process of the sale.

      Or…

Our application exhibits a slowdown during the checkout process of the sale. The CPU is spiking to 90% utilization and the complete Sale method of the Purchase class, which is executing on the middle tier of the application, is taking 12 seconds to execute.

The answer should be obvious. As a QA professional, your job is to provide as much information as possible to development in order to fix the problem.

Until now though, that information has been hard to come by. Traditional performance testing tools have only been able to identify the presence of a problem. Only today, with Rational Performance Tester's Root Cause Analysis feature, can you now pinpoint the cause.   With Root cause analysis, Performance Tester will

Identify slow code at the method level
Identify overloaded hardware resources such as CPU, Disk, Network, etc…
Identify application response time at any tier of a multi-tier application

Bottom line, you can now give your developers the information they need to fix performance problems.

Let's see how this is done.

<01_Schedule>

Using Performance Tester, QA Professionals can quickly record performance tests without having to write a single line of code. Groups of these tests are then assembled into test Schedules for execution. Among the many flexible options available at the schedule level are resource monitoring and response time breakdown. Together these two options provide all the information required to pinpoint a problem be it hardware or software related.

With resource monitoring, a tester can elect to monitor any performance counter available on Windows or Unix. Monitoring can take place on any physical tier of the application. Collecting this resource information makes it possible for users to diagnose performance problems that are caused by hardware resource utilization issues.

But what if the problem is not hardware related? What if the problem is software or code related? To pinpoint software or code related slowdowns, we can use the response time breakdown feature. This feature will breakdown a response time for a given transaction

by the type of code executed, and by the application tier where the code physically ran. It will provide detail down to the class and method level of code, to give developers a starting point for their problem resolution process.

<02_Analysis>

Performance test results are available both during and after the schedule execution. Here we can see that performance tester has indeed identified that a performance problem exists. The response time for the checkout page of our J2EE application is almost 15 seconds! With the problem identified, let's use the Root Cause Analysis features to diagnose the cause.

Let's first check for hardware issues. The Resources report graphs all selected resource utilization, and while most of our resource consumption is stable, we can clearly identify a spike in our CPU processor utilization.

Let's overlay this information with some response time information. By examining both response times and resource utilization spikes, we can look for cause and effect relationships. Here we do indeed see that our CPU spike occurs just prior to some of our longest response times.

Moving back to our page performance report, let's break down our response time by code and tier, to diagnose any poorly performing code. Our application is running on the "demo" tier, and by sorting the response times, we can see that our biggest slowdown came when we were executing some JDBC code, specifically the getConnection method of the JDBC class.

Information at this detailed level is invaluable to developers and saves them countless hours of trying to reproduce and debug performance problems.

<03_Conclusion>
Rational Performance Tester's Root Cause Analysis takes you beyond problem identification to problem diagnosis. It will identify resource constraints and poorly performing code saving your developers from countless hours of frustrating performance problem research.

If you'd like to learn more about this or any other feature of Performance Tester please visit our web site and download a free evaluation version of the software.