

View Point



Enabling Performance Testing with Functional Test Tools An Innovative and Inexpensive Approach

Dick Van Driel, ABN AMRO Bank
Surya Prakash, Infosys

Abstract

Typically QA teams face two major challenges during the performance testing phase: Procurement of expensive tool licenses and unavailability of a newer/compatible version of the performance testing tool. However, it is imperative for QA teams to come up with innovative and cost efficient solutions to overcome these challenges and ensure applications and systems are tested for performance and that quality is not compromised.

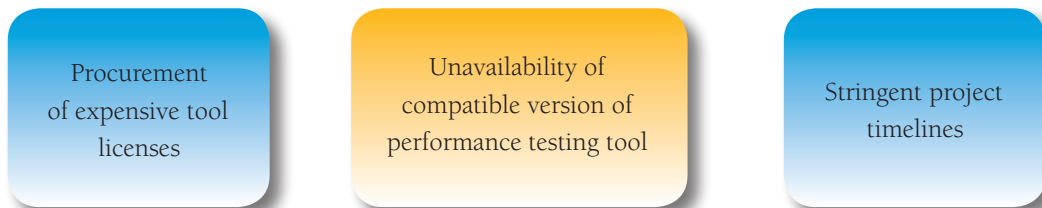
The following document, based on actual experiences, introduces one such innovative approach to performance testing. It outlines how performance testing challenges can be overcome using existing resources – such as functional testing tool in place of a performance testing tool, and terminal servers or virtual desktops to simulate the load requirement.

PERFORMANCE TESTING: THE NECESSARY EVIL

Unlike yesteryears when IT's primary role was limited to ensuring smooth operation of back-end applications, IT today also plays a crucial role in running and supporting revenue generating platforms (Ex.: E-commerce as a main channel of sales). This has compelled organizations to test the performance of their systems thoroughly, lest the slow or faulty software applications turn away existing and potential customers for good. As IT systems have evolved so too has performance testing, which may now be considered a necessary evil. Ensuring top-notch performance of IT applications may not boost your revenues but failing to do so will definitely impact revenues and reputation adversely.

Challenges encountered by Quality Assurance (QA) teams

In our experience we have noticed that QA teams are typically hindered by the following factors while testing the performance of applications:



While usage of test tools boosts quality and productivity of QA teams significantly, it also requires huge initial investment – in purchasing tools and licenses. QA teams not only need to purchase licenses for functional testing tools, but also purchase licenses for performance testing tools separately.

Another problem that QA teams grapple with is the unavailability of a version of the performance testing tool which is compatible with the applications and systems being tested. At times the performance tool is unable to recognize certain screen elements used by newer applications and this leads to errors during execution of the test scripts.

So, aren't there any workarounds to the above mentioned problems? Well, the incompatibility issues can be solved by sharing the code for the new application with the company developing the performance testing tool. The company would then help develop a patch which is compatible with the code. However, as one may imagine, most organizations are uncomfortable with such an approach due to security apprehensions. An alternative is to engage consultants from the company responsible for developing the tool. Again, this too is not favored by organizations since it entails expenses.

Short and stringent project timelines also play a role in eliminating the alternatives outlined above, since hiring consultants or developing a new patch for the performance testing tool would require considerable time and effort.

So is there a more practical solution that QA teams can adopt to overcome these challenges? There actually is! QA teams may consider using functional testing tools for performance testing too. Performance testing primarily consists of 2 elements – load generation and monitoring system/application behavior. While monitoring of system/application can be handled using existing or easily available utilities (PerfMon, GMon, inbuilt Unix commands etc.), functional test tools can be used to simulate generation of load. In fact quite a few parameters can also be tracked using the functional test tools themselves.

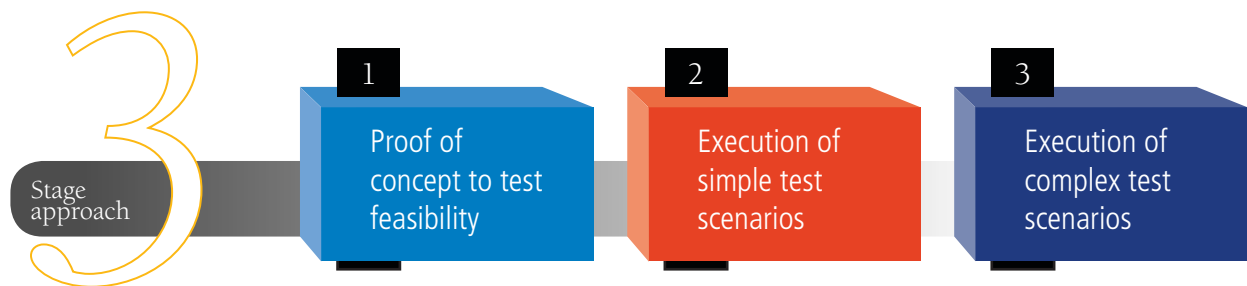
However the teams need to know their priorities. The approach we are going to propose works well in case of applications where the primary purpose of performance testing is to ensure optimization of Databases, response to data inputs and, CPU and Memory utilization of servers. Certain characteristics of the application lend themselves to this alternative better than others. Some such characteristics are:

- Number of users who will be accessing the application simultaneously is not very high (around 300)
- The overlap between functional test scenarios and load test scenarios is high
- Applications developed are based on object oriented concept which allows functional test tools to recognize screen controls

Functional testing Tools - Innovative and Inexpensive Approach to Performance Testing

As surprising as it may sound, functional testing tools can in some cases be used for performance testing as long as they can recognize screen elements. This is a more practical approach since all QA teams would already have functional testing tools in place, thereby eliminating the entire procurement issue.

Based on our experiences we have outlined below a 3-stage approach to test the performance of the applications effectively using functional testing tools –



1
Proof of concept to test feasibility

Functional testing tools can be used to perform the proof of concept to ensure the solution's applicability to the situation at hand. Performance testing tools come with what are called "load injectors". These load injectors help simulate load. While using functional testing tools for performance testing, we can make use of Terminal Servers as "load injectors". However it is important to test the performance of the terminal server during the proof of concept stage itself. The performance of the application to be tested is first benchmarked by executing a sample test scenario using 5 desktops individually. Next the terminal server is introduced to check if there is degradation in the performance of the system. This involves loading the functional test tool on to the terminal server and accessing it through individual desktops to execute a simple test scenario. In fact the 5 desktops can be simulated by opening 5 concurrent sessions through a single desktop. Further a simple code can help record the time required for completion of each activity or transaction (sample test scenario) and collate the data in to an excel sheet for later analysis.

2

Execution of simple test scenarios

QA teams can segregate test scenarios into simple test scenarios, requiring concurrent completion of a single type of transaction, and, complex test scenarios requiring concurrent completion of different transaction types. The performance of the system is tracked using different monitoring tools. Most types of servers which are likely to be used as terminal servers have inbuilt utilities which can be used to track usage of CPU, memory etc. For instance, Windows based servers come with the utility "PerfMon" which helps track these basic parameters. Similarly, the application server can also be used to track the same parameters using inbuilt utilities. Additionally, Java utilities loaded on to the application server help teams track important parameters such as duration and frequency of garbage collection. This is helpful in identifying any bottlenecks in the system which might otherwise slowdown the performance of the system. In case teams realize that the capability of the terminal server is creating a bottleneck, a simple workaround is to use more terminal servers.

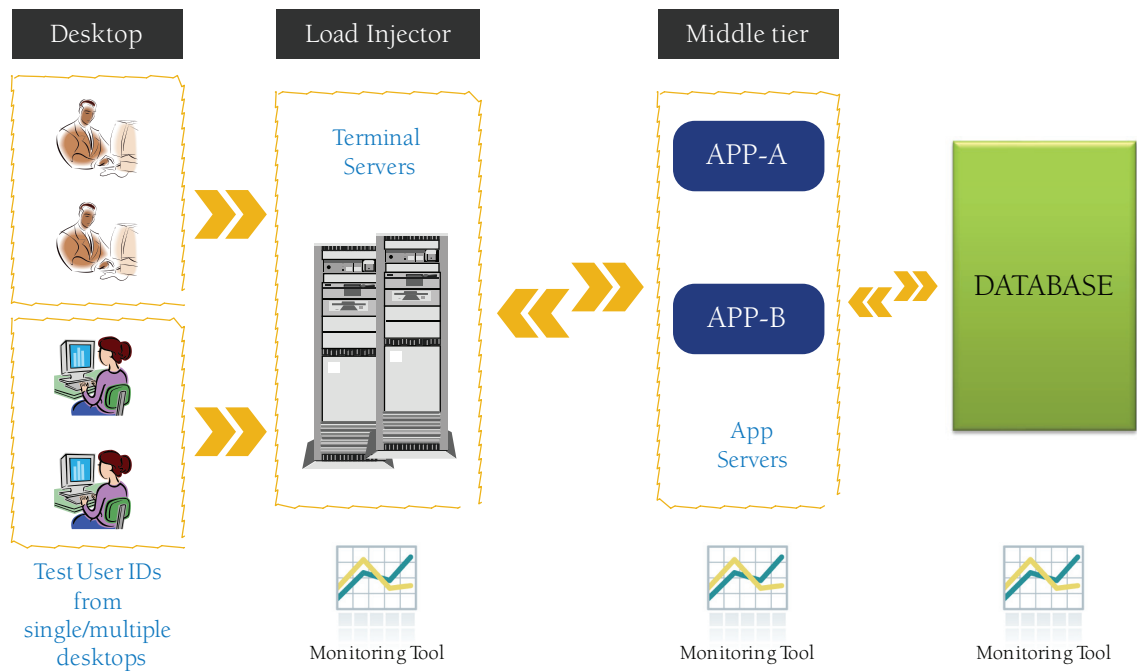
3

Execution of complex test scenarios

Complex test scenarios which involve execution/completion of different types of transactions concurrently may need multiple terminal servers. The decision to use multiple terminal servers depends on the capacity of the servers, namely the processing speed and memory. In some cases, especially during execution of complex test scenarios, the business team may not have explicitly shared performance requirements. However, the QA teams can proactively test the systems and help the business team set performance benchmarks. Consumption of resources is tracked as described above in the execution of simple test scenarios.

At the end of each test cycle, macros are used to collate data from different excel sheets and validate the performance of the system or application being tested.

Solution Setup



Usage of any automation tool involves the purchase of licenses which add to expenses. The cost of the license will depend on the type of protocol and number of users. However, in case the team already has a large number of licenses for the functional tool, then, by our estimates at least 20% of the cost of testing can be avoided by eliminating the need to purchase licenses for performance testing tools.

Most often testing the performance of systems requires dedicated servers to inject/simulate load. Such servers need to possess high processing capabilities and memory. Needless to say procurement and maintenance of such servers is expensive. By adopting this innovative approach to performance testing, hardware costs can be reduced by up to 30% by avoiding the procurement and use of expensive load injectors.

Additionally, reuse of the functional automation scripts helps reduce scripting effort by almost 60%. The scripting languages used by functional and performance tools are different from one another. Hence, the effort required for scripting performance test scenarios is independent of effort required for scripting functional test scenarios. In the approach outlined above, the scripts used for functional testing can be tweaked slightly and reused during the performance testing phase.

While the approach outlined above has distinct advantages, it does have limitations too. One of the key constraints is that the load simulation capability is primarily dependent on the capacity of the terminal server. Further, the approach of using functional testing tools for performance testing enables measurement of only key performance parameters (CPU and Memory utilization, Database performance, garbage collection rate and response time of application). While in some cases this may suffice, in other cases the tracking of additional and advanced parameters, such as Cache byte per second, Incoming Packet Rate, Page-in Rate etc. may be essential. Hence, in such cases one would definitely require the use of specific performance testing and monitoring tools.

Benefits



Conclusion

Having adopted the solution ourselves, we believe that using functional testing tools for the purpose of performance testing provides a great alternative to either engaging expensive consultants for developing a new patch for the existing performance tool, or procuring expensive licenses for the performance testing tools. We strongly recommend that all QA teams should definitely explore this approach before embarking on their performance testing journey, since the benefits of faster time-to-market and reduced cost of testing, when compared with traditional alternatives, are compelling.

About the Authors

[Dick Van Driel](#), Project Manager, ABN AMRO

Dick Van Driel is a Project Manager with ABN AMRO and is responsible for managing the separation and integration project within FX & Rates business. He has specialized in complex administrations, IT implementation projects and re-organizations. Dick has also been managing a complex project for implementing a new Trading Platform within ABN AMRO.

[Surya Prakash](#), Group Project Manager, Independent Validation and Testing Services, Infosys

Surya Prakash is a Group Project Manager at Infosys' Independent Validation and Testing Services practice. He has a keen interest in Performance Testing and has spearheaded many performance testing engagements. Prior to joining Infosys, Surya Prakash worked as Project Manager for a multi-national company and was responsible for implementation and testing of software services.



For more information, contact askus@infosys.com

About Infosys

Many of the world's most successful organizations rely on Infosys to deliver measurable business value. Infosys provides business consulting, technology, engineering and outsourcing services to help clients in over 30 countries build tomorrow's enterprise.

For more information about Infosys (NASDAQ:INFY), visit www.infosys.com.