



Performance: Still the Cinderella of the Testing World?

Last Updated: 20th August, 2009



Introduction

Over the last decade or so the importance of software testing has grown enormously. This has been reflected in the massive investment in people, training, methodologies and tools. Such investment is now seen as critical by many organizations in today's highly competitive world.

While it has never been more important to ensure that the end user experience is a positive one, there is one key area that is still often neglected: performance testing. This is particularly surprising given the greater risk and prominence of failure. For example, an ever increasing number of E-commerce and self-service applications are being deployed on the Internet based on new technologies such as virtualization and Cloud Computing.

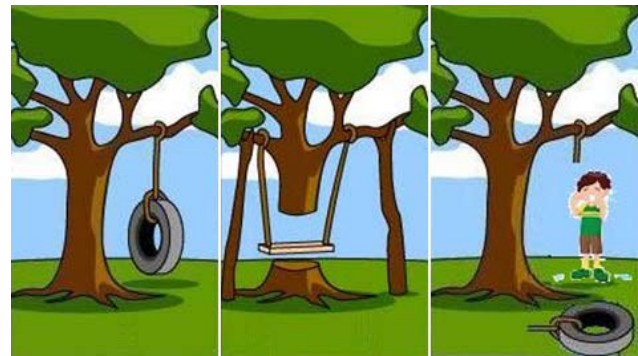
Despite the above, software is still developed against vague or even absent performance requirements. Where requirements do exist, little or no time has been allocated in the project plan for testing them. A similar story applies to the budget set aside for the people and tools and test environments required.

This document highlights some key areas where organizations can improve their performance testing practices and thereby reduce the risk of a negative end user experience.

Getting the Requirements Right

Many of you have probably read the Tree Swing analogy (http://www.hickorywind.org/images/Software_Treeswing.jpg) which highlights the importance of requirements and ensuring that an application is delivered against them. As a reminder, below are two key illustrations from this analogy followed by a third which highlights the added importance of getting the performance requirements right.

The pictures show that delivering a functionally correct system is not sufficient if it literally fails a load test and breaks.



What the customer really needed

How the Analyst designed it

What happens without a Performance Test

The key to performance testing is getting the requirements right and defining them early in the project. In our experience, this is often made more difficult because new systems and technologies have no predecessor on which to base expectations. This is where an experienced test partner such as AppLabs can help early in the life-cycle by ensuring that:

- ▶ The relevant stakeholders are involved in identifying and agreeing the performance requirements for the system. Relevant parties can include development, marketing, business analysts, system architects, application support and management. The independence of a test partner can help facilitate discussions and agreement in these situations.
- ▶ The requirements are captured in a specification document and signed-off by all parties concerned.
- ▶ The requirements are defined in an appropriate manner. For example, it is not best practice to define performance requirements in terms of average response times. Based on a 5 second response time for an application, if 50% of users experience a response time of 1 second, the requirement is satisfied even if the remaining 50% experience responses in the region of 9 seconds. Such behaviour could be caused by a badly performing server behind a load balancer. Imagine the potential loss of customer confidence and loyalty if the system was released.
- ▶ The requirements take into account the competitive nature of your market and users expectations of continually improving performance.
- ▶ Opportunities are assessed for performance testing early in the life-cycle. It is often beneficial to performance

test key sub-systems or components before they are integrated into the overall system and it is too late to fix any serious problems.

Planning for Performance

While the vast majority of projects will try and plan the development and testing activities early in the project, relatively little attention is typically paid to non-functional testing, especially performance testing. There are probably several reasons for this. Despite the fact that performance testing tools have been around for over 10 years now, there is still relatively little practical experience in today's IT world. Performance testing also occurs very late in the life-cycle and so the time allocated to it is often squeezed.

However, unless adequate time and resources are allocated for performance testing, the risk of delivering a "broken swing" is much more probable. In our experience, a performance test typically requires:

- ▶ 4-6 weeks (sometimes longer) in order to define, develop and execute a variety of test scenarios.
- ▶ An experienced performance test consultant (not just a tool smith but someone who can liaise with your business at different levels).
- ▶ Investment in a suitable tool that has been proven to work with your application (rental can be the most cost-effective route).
- ▶ 1,000's or even 100,000's of pieces of test data in order to feed the scenarios which may run for several hours (for example, a soak test may run for 8 hours or even longer).
- ▶ A dedicated test environment that is representative of production in terms of scale and specification. If such an environment is not provided for the exclusive use of performance testing, the prospect of sharing with others has to be factored into the test plan and timescales.
- ▶ Contingency if performance problems are discovered. In our experience, the majority of performance tests uncover issues which can take several days or even weeks to resolve.

Performance Monitoring and Closing the Loop

After a system goes live, those first few weeks or months are critical to its success and the end user experience. No

matter how thorough and well planned your performance tests, there are bound to be situations which have not been foreseen which can cause performance issues in production. This is much more likely in the case of a new system where there was no precedent or experience on which to base the performance requirements and tests.

AppLabs advocates the use of a performance monitoring technology to complement the use of pre-production performance testing. This will alert you to any issues and may in some cases allow some pre-emptive action. Provided the right type of reporting is available, performance monitoring tools can also provide you with intelligence on usage patterns in the production environment which can be fed back into performance tests of later versions of the system. For example, assume one of the scenarios simulated a maximum of 50 users logging into the system per minute. If in real-life the system experienced a maximum of 120 per minute in the run-up to Christmas, the scenario could be amended accordingly to reflect seasonal limits.

Conclusion

While testing practices have improved generally over the last 10 or so years, performance testing is still neglected in many projects due to time pressures and a lack of investment and experience. Yet there has never been greater pressure to provide a competitive end user experience, and a greater risk of failure due to the take-up of new technologies like virtualization and Cloud Computing.

Organizations simply cannot afford to release a poorly performing system and suffer the consequences in terms of customer loyalty. By working with an experienced test partner such as AppLabs, the risk of developing and releasing such a system can be greatly reduced by:

- ▶ Identifying and defining the performance requirements in advance
- ▶ Planning the performance test and investing in the right tools and people to make it a reality
- ▶ Monitoring the actual usage of the live system to ensure that users experience good performance and that future performance tests are fine tuned to reflect real-world usage.

Gary Wilson is Principal Consultant for AppLabs, the world's largest software testing and quality management company.