Rational. software

The Key to Successful Automated Testing: Planning

The complexity of modern client/server applications has elevated testing to become a critical and essential part of the development process. Today, no one would consider (or admit) not planning their software test efforts. Yet studies and surveys indicate that test planning is often a low priority item when it comes to software development. Worse yet, it is often not performed, or when it is, it's not complete, not accurate, or not followed consistently.

Assuming we all agree that testing is necessary, the questions we must answer next include: How do we test? What's really involved? How can we be sure we've done an effective job and really improved the quality of the application?

The answer is simple:**planning**.

This is the first in a series of articles that will review the role of test planning in the software life cycle and the concepts of effective testing planning. In this article we'll discuss why testing, particularly automated testing, is required. Then we'll introduce the concept of planning: why is planning so important? In later articles we'll break down the various elements of test planning, and offer some insight into how to tackle the process to maximize the chances of success.

Why spend the effort to plan?
The results of poor test planning are more visible today than ever before. Examples abound -- just look at a newspaper or magazine. A few of the more visible examples of poor quality software include:

- **AT&T** - A software switching system failed and hindered long distance communication in the US for almost 24 hours. The resolution required a change to a single line of code.
- **Denver Airport** - Software defects delayed the airport's opening almost nine months, at an estimated taxpayer cost of about $500,000 per day.
- **Ashton-Tate** - In the 80s, Ashton-Tate's DBASE software was the ipso-facto standard for PC-based database applications. Buggy releases resulted in diminished earnings and ultimately Ashton-Tate's (and DBASE's) demise. Ashton-Tate was ultimately purchased by Borland (who had a competing database management application, Paradox).

Of course these were all high-profile companies and projects. Not in your league, right?

Wrong! We all face software defects that may be as critical and visible within our organizations as these were to the outside world. Here are some of the more common symptoms of poor software quality:

Productivity losses

- System performance
- Lack of functions / features
- Does not meet user's requirements - lost sales

User frustration

- force users to perform tasks in non-intuitive manner
- work arounds
- delays
- does not meet expectations
- allows for user mistakes and misinterpretations

System outages, data loss / corruption

What's so different about Client / Server Applications?
Client/server applications offer various challenges to Quality Assurance professionals, below are some of the more important:

Rapid Application Development
Most client/server applications are developed using a Rapid Application Development (RAD) methodology. Testers must "try to keep up" with these short development cycles. Earlier, non-client/server applications often took 18 - 24 months for full development and initial deployment. Now, with RAD, applications are delivered in multiple deployments or "chunks." Each chunk is based on the previous one and includes enhancements, modification, and fixes. Each chunk requires multiple builds or iterative prototypes. Each chunk requires testing, and all within the shorter, 3 - 6 month time frames!

Client/Server Architecture
Today's client / server applications require many software components to function together, including the client application, the workstation's operating system, the network, and the database management system. There are often other components, such as the DLLs (dynamically linked libraries) containing additional code for proper execution, transaction processors, or application and database management services. Each additional "layer" of software adds additional complexity (and the need for testing) to the client/server architecture.

Multiple Types of Testing
Additionally, testing a client/server application requires many different types of testing, such as functional, user-interface, performance, and configuration testing to name a few. Each of these tests focuses on one or several test objectives. To prevent testing from wandering or trying to test everything simultaneously, each must be carefully planned. This is especially true when automating your testing.

Data
For each type of test we are executing, data must be used. Data is crucial to the execution and success of testing, as it will be used to identify the initial application data state (condition) and invoke or elicit a specific event or action. And data is used to verify that the tested event or action occurred properly!

Additional reasons to plan for testing
As discussed above, today's applications are very different from applications developed in the past. Client/server technologies have increased our ability to develop and deploy mission-critical, enterprise systems, often with shorter schedules and increased functionality. Client/server applications also provide developers and end users with a multitude of options and control. But each of these benefits bring with them increased demands on testing.

Test Automation
Increasingly, software testing must utilize test automation tools and techniques to meet these challenging schedules. But tools alone are insufficient, successful test automation requires test planning. Implementing test automation without planning is automating chaos. Using test automation tools, we can manage chaos and identify those elements of the process that contribute to it and add overhead to a project, (such as "undocumented features/changes").

Test automation is the only way software testers can keep up with the developers and test each new build with the same level of confidence and certainty as previous builds.

But testing often challenges testers for the most efficient and productive use of their time and efforts. Test automation introduces a new resource requirement - test development. Manual testing required test design to identify what and how to test, but since no tools were used, there was no need to develop any test scripts or procedures, just set up the system and start using the keyboard! This is not an efficient use of time, when you consider that for every test that needs to be executed, a resource is needed at the keyboard.

Test development is a new skill set required to use the tools and generate the test procedures after they've been designed. Effectively, three different individuals could be used, enabling you to keep your most senior resources on the design and planning tasks, while more junior resources (or outside resources) are used for the development and execution. This allows you to augment your staff when needed and share resources, with fewer disruptions to project schedules.

Defect Management and Analysis
Defects will be found. That's the objective, or goal, of testing, so we must identify and communicate the defect life cycle and analysis of results to ensure that defects are handled effectively and efficiently. Test planning ensures that defect management and analysis is an asset to the project, not an impediment. If you don't have an existing defect management system and process, or one that works well, test planning is your opportunity to define (or revise) it.

Test planning should also identify what metrics will be used. Test planning addresses how well you measure product quality. It also addresses how defect density or defect trends are measured and communicated.

Additionally, test planning can identify and communicate how data is collected and distributed, should also be identified and what report formats will be used, and when will the reports will be made available.

Risk Analysis
Test planning provides the opportunity to assess risk. Risk and liability can be a nightmare to the organization, but they can be managed. However, they first must be analyzed. Risk analysis helps to prioritize and focus our test efforts, ensuring that the right things are tested, for the right reasons, in the right order. (Right being based upon risk and acceptability to the organization.)

For each project, a risk assessment is performed and used to identify the potential risk or liability of an undiscovered defect. Risk should be used to assess the impact of a defect to the immediate end-user, data, or other end- users and applications. This data can be used to establish test priorities and assess any constraints, such as time-to-market, budget or costs, or quality concerns.

A review of existing standards, guidelines, and requirements should also be included as part of the risk assessment. The purpose is to analyze these documents, their appropriateness for the project, and implement or revise them accordingly.

It is also important to review any outside influences that may affect or impact your project. These influences might include specific user requests, regulatory requirements, or market conditions, any of which might change your assessment of risk or priority.

Process Improvement
Test planning is your documentation of the test process. Planning your testing is not only your opportunity to document and communicate the test effort, but also allows you to review the effectiveness of your test efforts.

Have you ever heard any of the following:

"The users reported a defect in ..., didn't you test this?"

"This was so obvious, how could you release a product with a defect like this!?"

"I know you stated you needed three months to test, but you've only got two..."

Improving product quality (fewer software defects) requires that the process by which the product is developed should undergo constant improvement. Developing a test plan enables the testers to identify, execute, measure, and improve upon their test efforts.

In summary, well planned, automated testing is required for several reasons. First, organizations that don't test significantly increase the likelihood of major system failures, resulting in delays, expensive fixes, and potentially irreparable damage to customer confidence. Second, the nature of modern client/server applications permits rapid development of highly complex systems that simply can not be adequately tested using traditional manual methods. Finally, planning is required in order to manage the accelerated testing process, analyze and track discovered defects, perform critical risk analysis, and to constantly improve both the testing and the development process. In the next article we'll take a closer look at the planning process, identify the specific steps involved, and discuss how to develop an effective startegy for the planning process.

IBM software integrated solutions
IBM Rational supports a wealth of other offerings from IBM software. IBM software solutions can give you the power to achieve your priority business and IT goals.

- *DB2® software helps you leverage information with solutions for data enablement, data management, and data distribution.*
- *Lotus® software helps your staff be productive with solutions for authoring, managing, communicating, and sharing knowledge.*
- *Tivoli® software helps you manage the technology that runs your e-business infrastructure.*
- *WebSphere® software helps you extend your existing business-critical processes to the Web.*
- *Rational® software helps you improve your software development capability with tools, services, and best practices.*

Rational software from IBM
Rational software from IBM helps organizations create business value by improving their software development capability. The Rational software development platform integrates software engineering best practices, tools, and services. With it, organizations thrive in an on demand world by being more responsive, resilient, and focused. Rational's standards-based, cross-platform solution helps software development teams create and extend business applications, embedded systems and software products. Ninety-eight of the Fortune 100 rely on Rational tools to build better software, faster. Additional information is available at www.rational.com and www.therationaledge.com, the monthly e-zine for the Rational community.

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.

Printed in the United States of America
01-03 All Rights Reserved.
Made in the U.S.A.

The IBM home page on the Internet can be found at **ibm.com**