

Risk Based Testing: Deferring the Right Bugs

Risk and Testing:

Does more testing make the software more stable?

Even the unlimited project schedule and resources can not eliminate all the risks associated with going live. Accepting the release readiness of the system would mean accepting a certain level of risk.

Risk Based Testing: Shipping with known bugs

Effective traditional testing means finding the right bugs whereas risk based testing involves deferring the *right bugs*. This fact sets it apart from the strategies, benefits and limitations involved in the traditional form of testing.

This technique establishes the release readiness criteria not by the number of open defects but the consequence of these defects. It, thus, allows you to ship your product with known defects.

What is Risk Based Testing?

It provides a critical level of objectivity in determining what things to test by using a combination of business and technical requirements to prioritize the testing process. Though it's a problem based testing technique when applied in the case of looming deadlines and limited resources, but if combined with the traditional ways to test the application, it limits the scope of the testing efforts.

This technique makes it easy for the QA organizations to deliver the *right level of quality* as a response to changing business requirements.

How does it work?

Risk-Based Testing helps to find the *right quality level* that can be delivered in limited schedule and resources by identifying business and technical requirements for an application and prioritizing these requirements on the basis of likelihood and impact of failure.

This strategy involves setting risk indicators to prioritize the testing tasks by:

- Analyzing the most important/critical functionalities: Priority Indicator
- Analyzing the areas where the probability of failure is high: Defect Indicator
- Analyzing the impact of failure of a particular functionality: Severity Indicator

It's a metrics in itself that quantifies the progress of the testing project by providing a framework for test monitoring and control throughout the project. It builds accountability to all the tests performed thus bringing a clarity and objectivity to the entire testing process.

Risk Based Test Planning:

Formulating a risk based testing approach needs an understanding that there are risks involved before you start testing the product and there are going to be some residual risks that can not be eliminated. This assumption calls for a strategy to try reducing the likelihood of certain risks, categorizing the identified risks on the basis of their importance, designing tests that can eliminate the risks and devising a mitigation plan to reduce the impact of the risks that can not be eliminated.

Step 1: Review and Identify Gaps

Involve QA and testing from the initial stages of the SDLC rather than treat it as a distinct artifact of software development to reduce the probability of risks occurring at later stages.

Step 2: Identify Strategic Features

Identify the most strategic technical and business priorities. The purpose of risk based testing strategy lies in identifying and focusing on the most critical and strategic features. It's therefore important to understand the product's purpose towards both the end users (technical priority) and the value addition to the business (business priority). In business terms, these features can be categorized on how they are setting the product apart from the competition.

Step 4: Identify Risks

Classify components that can fail. Design test strategies that can test areas where probability of failure is high.

For example, modified areas are an important source of defect. Identify areas where any change in application is introduced (especially if following an iterative approach). Another such failure indicator is the area around the defect fixes. Defects have the tendency to bunch together in defect-prone areas as fixes lead to changes leading to new defects.

Step 5: Identify Risk Triggers/Source

Design tests that can explore the application from different perspectives, thus revealing the potential weakness of the application. These tests are designed with the intent to exploit the weakness if it exists. Merely identifying risk areas is not enough. Isolate the situations that can trigger that risk and cause a failure. Identify the ways to attack the capabilities of the software by forcing situations that are the cause of failure. This particular strategy is often termed as Negative testing. It can find significant failures and provide strategic information about the application under test.

Step 6: Assign the identified Risk items a Hierarchy

In the light of the risk indicators that have been identified, it's important to prioritize them on the basis of their priority (quantifies the necessity for the area to be prioritized for testing: *business focused*) and severity (quantifies the impact of a risk on functionality: *customer-focused*).

Setting Priority:

Areas can be prioritized on the basis of their criticality, visibility and frequency of use.

Setting Severity:

Areas can be prioritized on the basis of the impact on the user if a particular feature fails to operate correctly.

Step 7: Create Critical Path Test Cases

In order to make the risk based testing approach more exhaustive it's important to make the testing coverage more wide-spread. The test cases should be able to cover the basic flows with more emphasis on the critical flows.

Step 8: Devise a Mitigation Plan

The risk based approach not only targets the strategy that can expose all the risks and eliminate them, it also aims to reduce the impact of residual risks that were not anticipated or eliminated.

Knowing all the Risks: Risk Identification/Risk Assessment

Risk identification requires collecting all the system-related information that can be analyzed and classified to estimate the risks and the potential risk-sources involved. A complete understanding of the system and its operational environment helps to understand the potential risks thereby establishing a well defined scope of risk identification and assessment.

Some of the most probable sources that may trigger the possibility of risk becoming a problem are listed as follows:

- Unrealistic Project Schedule
- Incomplete or changing requirements that lead to New/Changed Areas
- Application size
- Application complexity: areas that are complex to understand and execute
- Component interdependencies/ Introduction of third party software
- Introduction to new technology/platform
- Project/Team Management Risks

Risk Mitigation:

All the risk management activities planned so far, i.e. risk identification and assessment give a picture of risks that can not be eliminated. For such risks, mitigation activities are planned to reduce the impact of the failure that can not be avoided.

- Releasing the product with recommendations
- Fixing the defects later in a reasonable timeframe (Patch release)
- Making the technical support ready
- Building Roll-Back plan

Optimize Risk Based Testing:

A risk based testing approach can be made more exhaustive by including few techniques like Beta testing, Automation and Metrics generation. This will facilitate the testing efforts and improve the testing coverage.

Include Beta Testing

If the software has been used before, an active user group can be helpful in testing new versions and expediting the entire testing efforts.

Add Automation for Test Management

To speed up the entire effort, Test Managers can automate the process of entering, tracking and resolving defects found during testing. Automation can also be used for correlating and tracing the results to the initial risk assessment.

Create Metrics to Gauge Results

A metrics can be generated to describe and analyze the relationship between risk and testing efforts. It articulates exactly what's been tested and not tested, giving accountability to the testing efforts. This set of metrics will be used to analyze the results against initial risk assessments, to make the relevant changes to the priority of testing and to re-evaluate the risk values of out-of-line results.

What if there is no prior knowledge of the product?

So far, our approach in understanding the risk-based testing was based on the assumption that we are testing a new version of the existing product and that we are familiar with the basic functionalities and purpose of the product.

What if it is a new product? What if all the risk indicators describe earlier can not be applied?

In such cases, we execute Exploratory Testing by exploring the use of the software within its overall environment.

A rough test should uncover all the critical functionalities, defect prone areas and few important failure situations. These features and areas can then be prioritized as described earlier.

Conclusion:

Risk Based Testing is an approach that requires skill and experience to isolate the most important tests on the basis of technical and business constraints.

Risk based testing is a powerful testing technique that enables the QA teams to streamline their testing efforts. Organizations that implement this technique are in a better know-how of the risks that are inherent in their applications and of the risks that are actually significant. This knowledge is much helpful in exploiting the relationship between risk and testing efforts, thus, bringing an objectivity to the test designing and test management. It allows QA teams to make informed decisions while setting a clear test exit criteria.