CASE STUDY

Test Asset Re-engineering

Re-architecting test assets improves test effectiveness and increases test coverage by 250%, resulting in reduced support issues and higher quality application releases, for a leading online banking solutions provider.

Domain - Banking/Financial

Technology - J2EE-based web application

CUSTOMER AND PRODUCT BACKGROUND

The customer is a leading provider of Internet banking solutions in the online banking and financial services domain. The product in question is an online banking solution that allows customers to control their finances through a single secure source website. The application has three different suites: bill payment, personal finance management, and funds transfer.

PROBLEM STATEMENT

The popularity of the application has seen it grow, in terms of both size as well as number of customers. The application was customizable, resulting in significant number of customization requests, which resulted in separate versions being maintained. The existence of multiple application versions caused numerous issues when it came to the related test assets, such as inconsistency, duplication, absence, redundancy, incompleteness, inability to adhere to standards, and being automation-unfriendly.

As a result, there were an increasing number of support issues and customer complaints, despite the presence of a strong in-house QA team. The customer realized the gravity of the problem and decided to bring in a test specialist to resolve the issue immediately and prevent the situation from going overboard.

SOLUTION

As a first step, a six-member team from STAG set up a good test baseline by thoroughly assessing the currently available test artifacts. These included functional specification documents, design documents, test plans, test scenarios/test case documents, and change requests for the quality of test cases, test completeness, and coverage and fault-finding ability. The thorough assessment helped in identifying the current gaps and chalking out an improvement plan by applying HBT techniques like Equivalence partitioning, Boundary-value analysis, Cause-effect graphing, Error guessing, and All Pairs.

- # Existing test cases: 4070
- # Test scenarios designed: 6403
- # Test cases designed: 7618
- % Increase in test cases: 87.15

STAG re-engineered the test cases using the HBT Test Case Architecture. This was achieved by first grouping the test cases on the basis of their features and then by levels, then segregating them into the various test types, and then finally sorting them into positive and negative test cases. This process of fitting the test cases into the HBT Test Case Architecture helped uncover an important fact: the number of existing test cases was inadequate. To resolve this issue, STAG designed additional test cases, 7618 in all, ensuring that it included more negative test cases to even up the mix. The team also ensured that test cases for displaying appropriate system error messages were also added. Finally, the new test case architecture ensured that future test cases could be plugged in easily.

For effective test case management, the STAG team introduced the practice of centralizing test cases to avoid issues of duplication and also of obsolete test cases staying on in the repository and adding to the overall test case count.

OUTCOME AND VALUE ADDITIONS

There was an increase in the test coverage caused by the uncovering of the missing test cases; it also provided a sharper visibility of the quality as the test cases were well organized by specific defect types. This improved the test coverage by over 250%.

STAG's solution also ensured total adequacy, easy upgradeability, and simple maintainability of the test assets.

Visit: www.stagsoftware.com | E-mail: marketing@stagsoftware.com
Bangalore: +91 80 28495574 / 75