EXPERIMENT - 7

Software Testing and Quality Assurance

Abstract

Study the Test Management Tool: QA Complete.

Syeda Reeha Quasar

14114802719

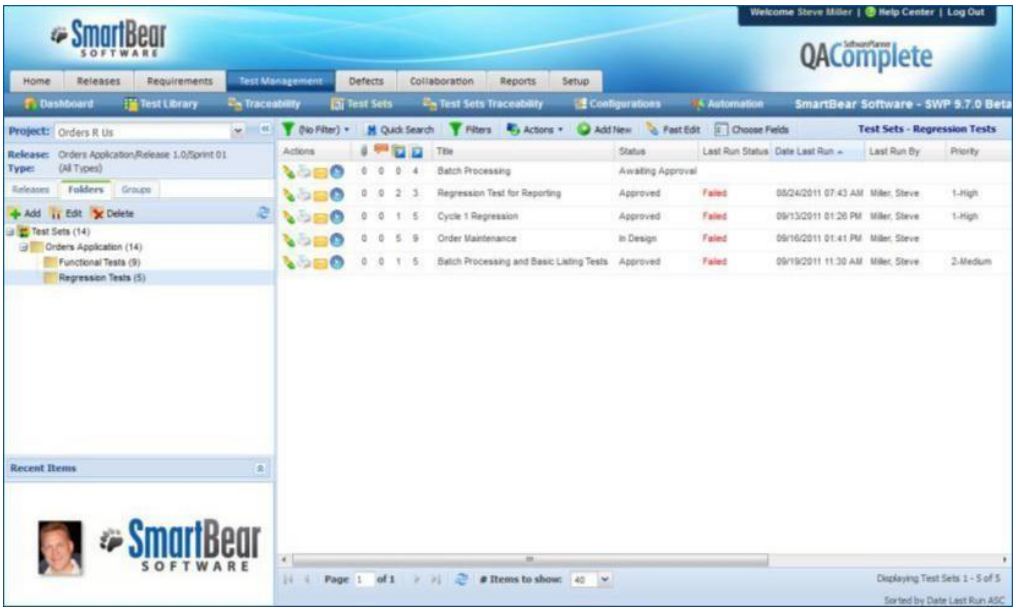
EXPERIMENT – 7

Aim:

Study the Test Management Tool: QA Complete.

Theory:

QA Complete gives the test team a single application for managing test cases, test environments, automated tests, defects and testing project tasks. QA Complete is the allencompassing solution, providing visibility of test management process and ensuring the delivery of high quality software.



QA COMPLETE FEATURES

- **TEST CASE MANAGEMENT:** Create test case suites to cover regression, smoke tests and standard release tests. Folder structure allows organization by release or feature. Simple test case structuring also allows for focused metrics and clear status reports.
- **TEST ENVIRONMENT MANAGEMENT:** Using List Manager track all test environments and configuration. Link environments to individual test cases to see how effective test coverage is across different platforms, operating systems and devices.
- **DEFECT AND ISSUE MANAGEMENT**: Track status and resolution progress of defects and issues for each release. Features, such as automatically creating defects on failed test cases, helps reduce time spent entering data and increase testers time spent running the tests.
- **TEST AUTOMATION INTEGRATION:** Integrators with test automation tools like QTP and

- Test Complete allow you to track and report on the whole test management effort in one place. With the ability to co-ordinate both manual and automated tests one can get the test data make release decision with ease.
- **BUG TRACKER INTEGRATION:** Integrators with Jira, Bugzilla and other web based defect tracking tools allow tying together test effort in QA Complete with existing defect tracking tools implemented within organisation.
- **TEST PROJECT MANAGEMENT:** Ensure that all test project tasks are tracked and monitored from writing the test plan to making the release decision. Variances can be tracked to help improve estimating and planning for future releases.
- **SHARED DOCUMENTS:** Avoid the usual mess associated with storing test artifacts on different servers, in different directories and in different physical locations. Store all test documents in one central location to improve collaboration and communication within the test department.
- **REQUIREMENTS MANAGEMENT:** Define requirements for each release and track the release each requirement is scheduled for. Workflow configuration allows tracking through user definable statuses. Integrated linking with test cases delivers clear requirements test coverage reports.

Result:

The QA Complete tool was successfully studied and explained.

Viva Questions

Q1. What is bug leakage and bug release?

A bug leakage results when a bug is detected which should have been detected in earlier builds/versions of the application.

A defect which exists during testing yet unfound by the tester which is eventually found by the tester/end-user is also called bug leakage.

A bug release is when a particular version of s/w is released with a set of known bug(s)/defect(s). These bugs are usually low severity and/or low priority bugs. It is done when the company can afford the existence of bug in the released s/w rather than the time/cost for fixing it in that particular version. These bugs are usually mentioned in the Release Notes.

Q2 What is the difference between build and release?

A “build” is given by dev team to the test team. A “release” is formal release of the product to its customers.

A build when tested and certified by the test team is given to the customers as “release”. A “build” can be rejected by test team if any of the tests fail or it does not meet certain requirements. One release can have several builds associated with it.

Q3. What are the automation challenges that SQA (Software Quality Assurance) team faces while testing?

1. Unrealistic expectations of automated testing
2. Using the wrong tools
3. Automate useless tests and neglecting important test cases
4. Choosing the wrong testing time
5. Lack of proper testing

Q4. What is Agile testing and what is the importance of Agile testing?

Agile test strategy supports DevOps and continuous testing. And continuous testing is important to improving product quality.

In Agile development, testing needs to happen early and often. So, instead of waiting for development to be finished before testing begins, testing happens continuously as features are added.

Tests are prioritized just like user stories. Testers aim to get through as many tests as they can in an iteration. Adding automated testing tools can help testers get through more of the testing backlog.

QA is everyone’s responsibility in Agile. So, Agile testers and developers need to work closely together. Communication and collaboration are key.

Agile development is often driven by tests. Developers use Agile testing methods like TDD (test-driven development) to write the test first. Then they write the code that will

be verified by the test. And developers and Agile testers should collaborate before user stories (e.g., requirements) are set.

Once development and testing are underway, communication remains important. Agile testers should be testing as developers write code. Plus, developers will probably do some testing. And Agile testers will probably do some coding.

In Agile development, the definition of done is a shared, standardized understanding among the team that a particular user story has been completed. The acceptance criteria in a user story are what will help drive the definition of done. If the user story passes the acceptance criteria, it can be considered done. This includes testing or validating the acceptance criteria. So, a test verifies that you've completed the user story.

It's important that both Agile testers and developers know what has been tested and what defects still need to be resolved.

Q5: Explain what should your QA documents should include?

Test cases are pretty simple – this QA documentation consists of 7 sections: ID, Test Case, Testing Steps, Expected Result, Status, Actual Result, and Comments.

1. **ID** is a unique number assigned to your test case.
2. In the **Test Case** section, you point out the requirement(s) you will be testing and provide a link to it in the specifications document.
3. In the **Testing Steps** section, you list all the actions needed to complete a test case.
4. In the **Expected Result** section, you summarize the outcome of a particular test if it is successful.
5. In the **Status** section, you indicate if a particular step passed or failed testing.
6. In the **Actual Result** section, you explain the outcome of a failed test.
7. The **Comments** section is not obligatory, but you can add it to leave some additional notes.

	A	B	C	D	E	F	G
1	**ID**	**Test Case**	**Testing Steps**	**Expected Result**	**Status**	**Actual Result**	**Comments**
2	1	Billing process	1. Select customer delivery	Delivery is selected	passed		
3			2. Print invoice	Invoice is printed	failed	Invoice cannot be printed	
4			3. Send invoice to customer	Invoice is sent	open		
5			4. Create open item in AR	New open item is created	open		