User Acceptance Testing of a software product

1.0 Software as a product

When you purchase any product which involve significant expense and is critical to business, you would evaluate the product thoroughly on its critical areas. For example, while you purchase a car, you would take a test drive to get the feel of the car on the road. At times, we also take services of a third party to independently assess the claims made by the product vendor.

We increasingly depend on software in every day business and every day life. As with any other product, what would appeal to a wide variety of customers gets developed as a software product. Software as a product performs or supports many business operations with pre-defined intelligence in the form of executable code.

Much of the complexity is hidden behind a seemingly simple user interface.

2.0 Software Product Testing Basics

As you make a critical and significant investment, it demands a thorough investigation whether investment would do justice to the cause. Just as in case of any other product, software also needs to be tested. After all, nothing can be more convincing than actually "seeing" and verifying the functionalities implemented.

Any good software product manufacturer would thoroughly test his/her product in its original form before release. Yet, he/she may not know your specific needs and environment. Even minor variations in the implementation environment could affect proper functioning of the software product Also, capabilities and problems with software is known quite often when you actually start using it, unlike more tangible products like automobiles. To ensure that it would suffice for and work well with your business environment and infrastructure, you would need to test with user's perspective, in a form and environment close to that it is proposed to be put into.

Apart from testing product various declared features and capabilities, it needs to be tested for acceptable performance, efficient resource usage, usability, maintainability, stability, robustness, and security concerns. While all these would apply to any software, relative importance would depend on the end users and the actual environment and use it is being put into.

3.0 The Challenges

Effective software testing requires good understanding of the domain and the product itself. It also requires good management and a thorough and objective assessment through a judicious mixture of "user-perspective" and "constructive-destruction" approaches.

Documentation on requirements, as it has evolved over a period of time, are often non-existent, insufficient or outdated. Often, priorities differ between various customers and users and product needs to cater to every one of them in tune with their priorities. Testing of the product needs to be done considering the specific usage and environments planned, as mentioned above.

Knowledge about underlying technologies also plays an important role in simulating similar test environment, diagnosing & reproducing problems and in test automation. This fairly good technical background though it does not have to be as much as a development team requires. But, it should be good enough to facilitate effective communication with development team.

As a human resource intensive, technical activity, software testing requires effective team and task management with proper planning, close monitoring and immediate corrective action.

As time goes by, product would evolve to cater to new demands and test efforts needs to be managed in this context.

4.0 Approach

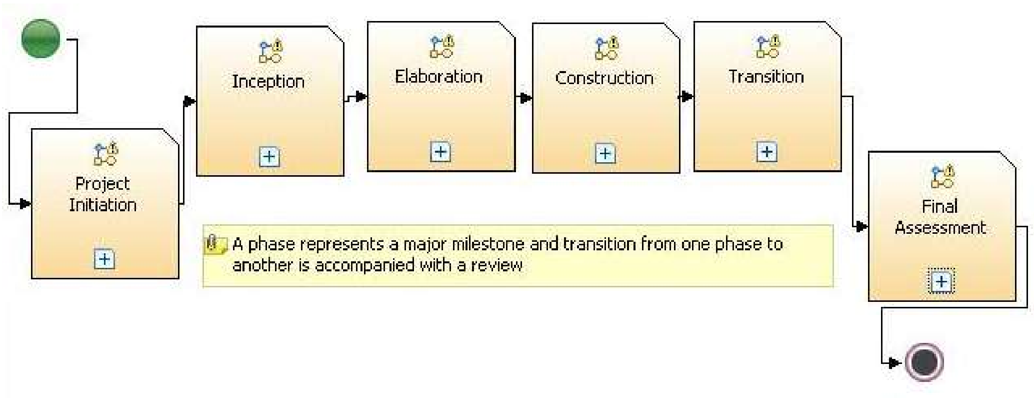
With an adaptive process, proper planning, effective communication, inherent support for risk management, close monitoring of progress and quality of work done, and immediate corrective action on feedbacks help succeed in testing the software. A generic approach for these testing is described below.

4.1 Process Model

As mentioned earlier, all capabilities of software are not equal in importance, ease of use or urgency of need. When an inventory module is rolled out, maintaining material masters and stock positions are expected to be automated immediately. But reports on optimizing stock advices can be taken only after a long period of use. This calls for intensive testing on what would get into immediate use. Similarly, placing an order in order processing module may involve payment through third party payment gateway. Testing this might require creation of a stub to simulate functioning of payment gateway as every test run would otherwise result in actual financial transaction. This call for conducting testing as a managed effort with explicit focus on risk, priority, effort involved, and effectiveness of approach.

Considering the above, Astra recommend an iterative model, analogous to the iterative model of the Rational Unified Process as shown below, for outsourced testing for user acceptance testing.

The overall period is divided into four phases called Inception, Elaboration, Construction and Transition with each phase consisting of short iterations. The phases represent major milestones. Iterations are short cycles of delivery enabling assessment, feedback and mid-course correction. Each of these iterations commences with a fine grained plan and evaluation criteria facilitating objective assessment at the end of iteration. Life cycle starts after the Project Initiation and closes with the Final Assessment, for persisting project knowledge and refactoring the project artifacts for reuse in a later project.


Project Initiation
Inception
Elaboration
Construction
Transition
Final Assessment
A phase represents a major milestone and transition from one phase to another is accompanied with a review

4.2 Best Practices

Having successfully resolved the software testing paradox in the industry, time and again, project after project, company after company, Astra has derived its own set of best practices

4.2.1 Adaptive and iterative process

Every product and every customer is different and a process which is well defined yet adaptive to specific situations is critical to success of testing project. As a human resource intensive, technical activity, success in software testing requires transparent functioning, proper planning, close monitoring of progress and quality of work done, effective & timely risk mitigation. Therefore, we recommend an adaptive and iterative process which renders itself to effective communication and control.

4.2.2 Manage requirements

Requirements are representation of concerns of various stakeholders. Managing requirements involve working with these from various perspectives and ensuring that the product is working in tune with these. Degree to which the product meets its immediate requirements as well as the evolving needs. Therefore, we recommend explicit focus on requirements

4.2.3 SMART goals

The project would be managed with an overall coarse grained plan as well as a fine grained plan for the current iteration. To get the best results for the effort, it is important to set the goals Specifiable, Measurable, Achievable, Realistic and Tangible.

4.2.4 Plan & Track activities

As a human resource intensive activity, software testing requires effective management. Well-defined plan should be clear on goals, approaches, entry & exit criteria, milestones, deliverables, resource requirements, and completion of testing. Further, work progress and deliverables should be managed against pre-defined milestones and shared promptly at appropriate levels among stakeholders. This ensures better transparency, effective communication, and smooth functioning across organizational barriers.

4.2.5 Manage Changes

All changes, including schedule change, resource (re-)allocation, defects, change in requirements etc, must be managed with an effective change management system with active participation of managers representing both the technical team and customers.

4.2.6 Manage defects

Defect needs to be considered not only as the result of testing but also, plays a major role in improving quality of the testing process itself. Defects needs to be traced back to faults and test plan should bring in test cases inspired from these defects and faults.

4.2.7 Effective documentation

Documentation should be kept light enough to keep it up-to-date on a continual basis yet must include critical information to facilitate communication.

4.2.8 Reusable test assets

As testing is an activity which needs to be done time and again for the product as it evolves, test assets once created should be kept as reusable as possible.

5.0 The unique value proposition

Astra understands importance of requirements in software testing and uses the best practices to document and manage requirements. Testing is performed balancing various stakeholders concerns and priorities as may be appropriate. Framework of well defined requirements in itself is a solid foundation for a software product testing.

Testing expertise and effort from Astra goes beyond functional requirements with specific focus on usability, performance, reliability, and integration aspects of software.

Testing efforts proceed with well-defined and practical plan, and focus on continuous risks focus and iterative process, with short feedback cycles, helps in early identification and resolution of risks like testability issues.

With iterative process, actual testing commences even before complete documentation is available giving a lead time and opportunity to bring early and continuous learning into management of the testing project.

Astra is a partner to, and has on-going relations with, leading software engineering and testing product vendors. It has certified and experienced consultants who have worked with wide range of applications and are well versed with various software engineering disciplines, like requirement management and software testing, as well as software engineering tools from leading vendors and open source. The collective experience would help implement the appropriate engineering practices quickly.

Wide experience of Astra consultants and explicit focus on test asset reusability would also help in getting better script resilience (using tools effectively) and heading towards next level of improvement in testing efforts.