Testing without Requirements - A Practical Approach

Anuj Magazine
anujmagazine@excite.com

Abstract

The title of this article may seem a paradox i.e. one may always argue that "How's testing possible without tester having knowledge of requirements". But, the reality is that most of the times, tester has to perform his job with minimum documented requirements or in worst case, no requirements at all. Such a situation not only makes tester's work difficult but also makes planning and estimation almost impossible and considerably reduces Return of Investment from QA department. In this write-up, I try to explore the situation of testing without requirements, its impact on whole Software process and how effectively a tester can perform his work.

Impact of testing with no requirements

Let's consider an ideal case in which a tester gets Software requirements well on time i.e. at Design and Analysis stage in the Software Life Cycle. In this situation- the tester's job will begin with reviewing the requirements. A lot of defects are uncovered at this stage as requirements are verified for ambiguity. This forms the base for creation of test cases. In the next stages, the test cases are derived from requirements and then execution of test cases takes place followed by analysis of results.

But the irony is that, this situation does not always exist for a tester. Testing is still considered to be simply as a separate phase in Life cycle rather than being accepted in each phase of life cycle. In this case, testing starts at the tail end of the Development process just few weeks before actual release. Testing in these conditions often makes life of QA department miserable and causes the below listed effects-

- As overall product knowledge is not documented in this case and is confined only to a few people, there is big task for QA i.e. extracting the information regarding product from developers, Product Management, Marketing and other concerned people. Most of the time these people are tied up with their work and unavailable. Testers hardly get to interact with them when most required.

- QA department's role in overall product life cycle becomes quite less. QA people are often spotted asking questions to concerted people regarding product's features. In this case, QA is often referred to as "**Q**uestions & **A**nswers" rather than "**Q**uality **A**ssurance".

- As testers are always chasing Product Management, Developers, this gives rise to unnecessary feeling to these people that they are superior to QA and hence respect for QA guys is compromised.

- Defect Migration increases i.e. the defects that were supposed to be found much earlier in the life cycle are found quite late. In this case, as the tester's knowledge regarding the product is gained over the period of time depending upon the clarifications received. By the time, tester gains good knowledge to test overall scenarios, the product is already nearing release. It is at this time most of the high priority bugs are found and lot is left for customer to find. This affects Return of Investment in a big way.

- Estimation of efforts, resources and planning in general becomes difficult. As requirements are not there, it is difficult to decompose the functionality and hence, the division of functionality becomes tougher and it changes a lot during testing life cycle.
- Most of the times, testers and Developers are out of sync regarding the functionality. It often happens that the tester logs a bug and developer

does not agree to it and there is a long argument between the two. As requirements are not documented properly, the perception of testers and developers regarding the functionality often differs. It can be best viewed in Defect Tracking System database. Often Bugs in this situation would have a long history of confrontations regarding the way, Software "should" work.

- The number of "AS-DESIGNED" defects in the defect database increases considerably. The tester logs the Bug but it turns out to be as per design. This may go against tester's appraisal, if management evaluates tester from number of valid bugs entered.

- Automation of test cases becomes infeasible. As test cases are created quite late, practically there is no scope of automation. The Huge investment, organization makes on acquiring the automation tools licenses goes waste, as there is no effective use of these tools.

Testing Approach

Despite the huge impact that lack of requirements has on the testing process and Software development process, tester do find himself in such a situation quite often. This forms one of the most common problems that testers face and is the root of many other problems associated with testing profession as such. Still, whenever such condition exists, the tester's resolve should be to do the best he/she can and manage the entire testing responsibility efficiently and effectively. The initiate should be taken to find better ways to do the work and provide a solid foundation for better process in future projects. Below are some of the points that help tester to work out the situation and perform the testing in an efficient manner-

Effective Oral communication

Communication is the most important aspect when the tester has to test the product. The information regarding product's functionality is available in scattered form with Developers, Product Managers and other departments such as Tech writers, Marketing etc. The tester's role is to efficiently gather all the information from the concerned persons and document it to maximum possible extent. As the information present would be in undocumented form, Effective communication will play an important role.

Arranging meetings at regular intervals with the concerned persons for clarifications of issues, queries form one important method. Such meetings are often of lesser duration due to high pressure of meeting deadlines, so communication from tester's point of view should be precise and only relevant points should be discussed. It has been observed that a lot of new test scenarios as well as bugs are also unveiled during such meetings itself, if handled properly.

Also, the tester should be very particular in asking queries, as this will determine his/her reputation with developers and Product Management. Asking obvious questions generally leads to hampered reputation.

Effective documentation skills

Tester may get a flood of information after interacting with concerned persons about the product's functionality. The next logical step is to organize the information so that it can be easily referred in future.

It is also observed that most people straightaway start testing with the information available without bothering to organize it. If the information is not organized- it cannot be effectively reused over the multiple test cycles. The information should be organized by documenting in a proper fashion. I firmly believe that a part of tester's job is to create the awareness of effective documentation in the organization. Proper documentation solves many a problems. Correct, precise and unambiguous documentation saves everyone's time and enhances productivity. Often, it is seen that testers show complacency and show their dislike for documentation. A tester needs to have good written and documentation skills

All the information that testing team gathers needs to be organized in the standard template and sent to concerned persons for review. Once reviewed, this documentation can be sued as a reference for creation of test cases and it can used by other departments such as technical writers. Note that such documentation is evolving i.e. it cannot be considered as complete even after couple of meetings with developers. Tester will observe that when he/she actually starts testing the product many new things will come up. All the newer clarifications should also be documented religiously.

The other important step in creation of effective documentation could be creation of flowcharts. Flowcharts depict the flow of control of a particular functionality. It shows the pictorial view of functionality under consideration. And, as it is said that "A Picture is worth a thousand words.", creation of flowcharts will make it easier to understand the functionality as well as find the missing links in understanding and more test scenarios are bound to come up. Flowcharts would again form the effective reference tool for writing test cases.

Other information sources

There can be other information sources- other than communicating with concerned people in the organization. They can be-

a. Dependency on external software
b. Information gathering from similar applications
c. Defect Tracking System
d. Technical knowledge

a. Any software application interacts with external software during its normal functioning e.g. An application interacts with Operating System (Win98, Win XP, Linux, Unix etc.) may be browser software (IE, Netscape etc.) or any other third party software it may have integration with. Integration with external software does form an important part of tests and should be taken care of in detail. It is worthwhile to spend time understand the integrating software. It will add to new test areas. These areas are often missed during normal testing.
b. Unless, the software you are working on is entirely an new idea, there are high chances that similar kind of application with (may be) lesser enhanced features is available. Testers can download the evaluation versions of such software and run through it. It will for sure open new avenues to test in the applications under test (AUT). It will also enhance tester's knowledge on competitor product and suggest enhancements to AUT.
c. In case, the product under test has been in existence for a long time and previous versions have been tested before (without requirements!), then defect tracking System's database can provide a lot of information on product's features. Tester can always refer to previously logged bugs and have a fair idea, which are valid bugs and which ones are invalid. It will always help the test team not to repeat the wrongly bugs.

d. Testers who invest their time in understanding the underlying technology of the product will have a better chance to understand the functionality in a better way. There would always be technical documentation that developers have created for their reference. Testers' having skills to comprehend these specifications and convert the information into good functional scenarios will prove to be a assets to the team. Having good technical knowledge will have other benefits also-
    - It will help to bridge the gap between testers and developers.
    - It will also help satisfy the people who have the misconception that "If you are not coding, you are not being productive".

Test Strategy

Test team will be loaded with many tasks at hand. As testers will be working under the pressure of meeting critical deadlines and ever-increasing pressure of finding bugs early, it is of supreme importance to prioritize the efforts. There may be the cases when tester is required to get clarifications, understand the product's features, document test cases and testing all at the same time. This is where the judgment of tester comes to play. There has to be a balanced approach between all tasks at hands and due importance should be given to testing the product. The whole idea is to uncover more and more high priority bugs as early as possible and be more productive.

Precise documentation of test cases is again subject to availability of time. But the testers needs to be ready with checklist which consists of –

a. All the tests to be conducted.
b. Proper sequence in which to execute the test cases. The correct flow will help to focus the thinking of tester at the time of execution of test cases.

The testing approach in this situation is often seen as Adhoc testing. In such form of testing, the tester randomly tests the functionality more on the gut feel. The problem is that on cannot always rely entirely on such form of testing as one is never sure of – "How much has been testing?" and "How much is remaining to be tested?"

This is where the Exploratory testing has a role to play. It is the more organized approach to testing. The features of software are explored thoroughly by the tester's thought process and more meaningful tests scenarios are tested and documented maintaining the proper flow of execution. The tester can best gain for such form of testing when he explore a particular functionality until he has tried all the ideas and then move on to next areas to test. This will help the tester to build the confidence in testing. It is very important to document all the ideas that were tested, as it will help in post analysis later on.

Another thing that can be of great help is what I would term as **Group Exploratory Testing**. In such form of testing, a group of testers explore functionality together. One person takes charge an explains to all, about the functionality and about the way he has tested it so far and plans on how he wants to test it in future. When a group of people explores the features from testing point of view, a lot of new scenarios comes up an infact may bugs are also unveiled. If offers multidimensional view of functionality.

Epilogue

No reason is a good enough reason for not documenting the requirements properly. This documentation more or less decides the productivity of concerned departments such as Testing, development, Technical Writers.
As a result of improper documentation, there is a huge impact on expectations from testing department, whole test process and most importantly, the Return of Investment that Management expects from the testing department as such. It leaves the role of Software testing to a mere formality rather than that of building confidence on product.

The message I would like to put forth is that if an organization is investing on the separate testing department, the best they can gain from it is when testers are involved in every stage of Software life cycle, right from reviewing of requirements till the release of software. It would ensure that everything is in place and quality of projects meets (and exceeds too) customer's expectations.

Acknowledgements

I take this opportunity to thank all my colleagues and superiors at my previous organization- Quark Media House (India) Pvt. Ltd. And the present organization- Network Associates Software (India) Pvt. Ltd. for providing me all the relevant experience.

About the Author

Anuj Magazine has an experience of around 4 years in the field of Software Product Testing and Process Improvement. He is a Certified Software Test Engineer (CSTE). He has published article at **www.Sickyminds.com** previously also.

He has been involved in testing CRM, ERP, Warehouse, Publishing, Network and Desktop Security related applications. He has been a part of Process Improvement Group in the Organization.

He started his career in Software Testing with Quark Media House (India) Pvt. Ltd. (Mohali), India and is currently working with Network Associates Software (India) Pvt. Ltd., Bangalore, India.