Thought of the Day: When a virtually flawless application is delivered to a customer, no one says how well tested it was. Development teams will always get the credit. However, if it is delivered with bugs, everyone will wonder who tested it!

\\\\\\\\Five most important things to remember to forget while testing are: \\\\\\\\\\\

1. Never underestimate your capabilities and different style of looking at a product

2. Never hide a bug from development team if it appears as a half bug to you. Half bug is something where you are not sure whether it is a bug or not. In any case of ambiguity or doubt in mind, just discuss the scenario with your peers, or superiors to conclude the things.

3. Cover the whole product but passing through each of the chunk. Your test cases have to work both at a broader level and micro level. The product sanity as a whole is as important as the functioning of each of its unit.

4. Deliver in time but without compromising with any of the situation.

5. Re-use test cases from your library where the repository keeps building up as a generic collection.

\\\\\\\\\\\\\\\\\\1. What is AUTOMATION\\\\\\\\\\\\\\\\\\

Automation is to execute your existing regression test cases without any human intervention. And your script should keep on updating stakeholders about the current status.

For good automation, manager and resources have to plan lot of things in advance, so that your automation suite is robust, maintainable and error free.

Following are basic points you should think of:

a. Automation framework installation in new machine
b. Proper documentation for each step needed by a QA professional to start suite
c. Easy to learn and easy to execute
d. Easy to add New Test Cases
e. Easy to Add new product
f. Easy to maintain Input data for different products
g. Good Logging for each test case, so that it can be used later if some error is found by automation.

h. Reporting to Stakeholder at each stage
i. scaling of your automation suite

- Increase in number of cases
- Increase in number of system

j. Script to check performance stats and Accuracy Stats

2. Automation Effort

There is an easy way to get automation effort needed to implement system in use. All the different steps can be divided into 4 high level verticals (Plan, Design, Test, and Maintain). Again it will depend how many automation engineers you have and how many machines you have to achieve your goal.
Think of PDTM when you want to automate any application.

PDTM stands for:

P: Plan
D: Design
T: Test
M: Maintenance

Below is the chart which describes 4 different levels of Automation:

2.1 Task Layout Plan:

1. Estimate Time
2. Define Scope
3. Create Test Plan
4. Form team, with different level of scripting knowledge on different tools needed in automation
5. Train team on different tools and different scripts integration
6. Finalize all features and test cases.
7. Create Final Schedule
8. Finalize on different environment your test case will run
9. Identify Test Data set.

Design:

1. Read Documents
2. Run Manual cases

3. Design Framework
4. Different feature and test cases can be selected as users will.
5. Review Framework
6. Design Reporting
7. Design Error Handling
8. Verify that different feature and test cases should be picked properly

Test:

1. Verify each test case
2. Verify test cases in batch
3. Run test cases with different dataset
4. Introduce defects to verify that your script catches those defects.

Maintenance:

1. Make Scripts version control
2. Make changes as and when feature changes
3. Maintain Script
4. Store script in secure location
5. Script should be updated with latest release

Automation Effort Graph

Overall effort calculation may have the following components:-

1. Test Requirement gathering & Analysis
2. Framework design and development
3. Test Case development (in case the available manual test cases not compatible)
4. Script Development
5. Integration Testing and Baseline.
6. Test Management.

2.2 Roles and Responsibilities

Manager

1. Estimate Time of automation
2. Understand Scope
3. Create a Plan
4. Form Team
5. Train Team with the product
6. Finalize features to automate
7. Create a Schedule

8. Check Environment Availability
9. Identify Test Data Set to be used for automation

QA Professional

Understand Phase:

1. Go through documents of application
2. Install and execute few cases manually

Design Phase:

1. Understand / Design Framework

Development Phase:

1. Write Code for Framework
2. Write code for Reporting
3. Write code for error handling
4. Write code for important functions in library

Coding Phase:

1. Code case and test case
2. Do thorough testing on each script
3. Run you script for several Test Data set
4. Introduce defect and check whether your script catches

Maintenance Phase:

1. Make your script as version control
2. Make changes as and when feature changes
3. Maintain for new environment and DATA set
4. Store scripts in secure location
5. Script should match with Application Release

How to Become a Successful Agile Tester:

Bangalore: A professional tester who can collaborate well with both the technical & business people and understand the usage of testing concepts to document requirements while driving development are known as agile testers. They are expected to have good technical skills, collaborate with other testers when automating tests. They are also expected to be well versed in exploratory testing.

Jessica Fleet in utest blog highlighted some tips provided by Brian Rock on how to become effective agile testers.

1. Focus on Risk/Change Based Testing
According to Jessica Fleet, the goal in an agile environment is to market a product within the shortest time without compromising on quality. They should have the ability to prioritize the tests based on the areas that have the greatest risk.

2. Understand the Architecture of the Product
A tester who has a good understanding of how the data flows through an application will not only be able to test the impact of any failures but also have the ability to deal with the potential security vulnerabilities.

3. Testing from a Customer's Perspective
In agile, testing should be based in the needs of the customers. Agile testers should remember that the fitness of the product for use by a user is the chief standard of quality. They are expected to understand how the end users will interact with the product.

4. Adapt to the Changes
The agile model allows changes to be made as per the requirements of the end user at any point of time. Thus ensuring customers' satisfaction. Agile testers should learn to cope up with these changes and provide quality assessments with regards to the quality of the product at any stages of the product's life cycle.

5. Leverage the Information in Application Logs
Agile testers should be able to leverage the information available in application logs as it can provide insights as to what the defect is and help in fixing the errors that can occur without the knowledge of the end user.

The Importance of Exploratory Testing In Agile Projects

Over the last two articles I have been touching upon various testing aspects in Agile projects - such as Test estimation in Agile, Test metrics in Agile. In this post, I want to touch upon a very interesting topic on the importance of **Exploratory testing** in Agile projects.

Quoted from Wikipedia "**Exploratory testing** is an approach to software testing that is concisely described as simultaneous learning, test design and test execution. Cem Kaner, who coined the term in 1983, now defines exploratory testing as "a style of software testing that emphasizes the personal freedom and responsibility of the individual tester to continually optimize the quality of his/her work by treating test-related learning, test design, test execution, and test result interpretation as mutually supportive activities that run in parallel throughout the project."

With such emphasis on the tester's individual creativity and freedom, this testing technique really brings out true potential of a tester. In the past, when traditional development models were more prevalent, the product development process was more rigid, with clear specifications laid down in detail. In the current day scenario where Agile projects have become more dominant, we all know that time is of essence and that team collaboration is encouraged more, than writing down all product and feature specifications. This is part in true because of time constraints but is also largely driven by dynamic product requirements, which make specifications obsolete very soon. All disciplines are de-emphasizing documentation and focusing more on hands on work.

Product specifications have been a guiding pillar for the test team all along in helping them understand the product, design test cases and know what to expect from the implemented piece of software. Given that this scenario is changing, where specifications are only written at a high level and not necessarily maintained throughout the life cycle, exploratory testing comes in very handy to help the tester understand the product's behavior and implementation. This helps them play around with the product in a free-flowing manner even before tests are designed, helping resolve any product queries upfront; it also helps the developers get a feel for product quality as quickly as possible.

In some agile projects, releases are extremely short, with time to market as short as 1 week or even less. It is not a practical use of the tester's time in such cases to go through the regular process of test planning, test case design, test execution, defect management etc. While test case design is invaluable, in situations such as these the focus on test execution (which is exploratory) is what will yield maximum return on investment. In all these situations the testers should not under-play the importance of test design and should spend time writing test cases for valid bugs from such exploratory efforts, to ensure they form part of a robust regression testing suite, for subsequent re-use.

Typically, the quality of bugs tends to be higher in exploratory testing rather than in formal test case based execution, due to two reasons:

a. **Exploratory testing** in Agile forces the tester to collaborate with the rest of the product team such as the development team, business team, design team etc. to clarify understanding in the absence of detailed specifications. Such collaboration promotes better understanding of the product, known issues, duplicate issues etc. forcing the tester to report only valid bugs

b. **Exploratory testing** encourages the tester to be creative in his/her own sphere, rather than be bogged by mundane testing focused around designed test cases trying to execute "x" number of tests per day. Such creativity promotes better end user role play and more realistic bugs rather than bugs that the team can afford to live with

If such is the power of exploratory testing, why has it not completely replaced the traditional techniques? Well, it has not, for the right reasons:

1. Mapping exploratory test efforts with the overall test strategy in making the important call of product sign off is often difficult. Challenge in establishing test traceability is a practical problem to address

2. Ensuring all the required types of testing are done including performance, security, localization, accessibility may be difficult to establish in exploratory testing. This is especially true when the tester goes into a comfort zone to conduct exploratory tests in just one area and loses focus on others. For e.g., a tester who is particularly good at accessibility testing may want to focus more on accessibility and usability aspects in his exploratory test efforts and not as much on other areas

3. Exploratory testing can get randomizing when used as a stand-alone technique especially for testers in their early stages of their career, not giving them a sense of confidence in their test efforts

Thus, while **exploratory testing** has its own benefits to offer especially in an Agile project, a test manager plays an important role in leveraging it in combination with formal testing techniques subject to project's constraints of time, cost, complexity and laying down areas that might additionally benefit from exploratory test efforts. Such an informed decision being cautious not to tress-pass into the tester's exploratory thought process will go a long way in reaping the benefits this technique has to offer in an Agile space.

Company Name	Key Person	Location	Brief Description
360logica Software Testing	Aman	Noida	Software, Outsourcing, Offshore, Manual, Automation, Black Box, White Box, bugs, error, database, System Integration testing, BETA, UAT, Test cases design
Cigniti	Raj Neravati	Hyderabad	Software Testing, Quality Assurance, Offshoring, Microsoft Gold certified Partner
Cityland Technologies	Srinivasa	Bangalore	quality support and maintenance services & applications in an efficient and timely manner.
Cybage	Arun	Pune	Software Re-engineering, Software Maintenance, Quality Assurance
Game Cloud	Lakshmikant B. Thipse	Pune	Game Bug Sweeping,Verification,Reporting, QA Testing Services for Games and Interactive Applications
Harbinger Group	Vikas Joshi	Pune	software accuracy & reduce cost of producing quality software, comprehensive testing strategy, quality & performance
Indium Software	Harsha Nutalapati	Chennai	System, Functional, Test Automation, Load, Unit, Dedicated Test Lab, Compatibility, Installation, Cloud, Game, Security, FDA, PCI DSS, Section 508 Compliance, Test Process Consulting
InfoStretch	Rutesh Shah	Ahmedabad	Mobile Handset and Certification,

			Mobile Application Infrastructure, iPhone Localization, App Store Certification, Product Engineering, QA and Process Optimization, ERP Testing
ITC infotech	Ashwani Maheshwari	Bangalore	key business domains, outsourced or independent testing, accelerate test cycles & improve quality.
Mahindra Satyam	C P Gurnani	Bangalore	QEdge, Mahindra Satyam's independent Product and Application Testing Practice
Maveric Systems Limited	Ranga Reddy	Chennai	Program Assurance, Process Assurance, Application Assurance
Mindfire Solutions	Chinmoy Panda	Bhubaneswar, Odisha	Manual testing, Automated testing, Regression testing, Load testing, Selenium, QTP, Mac Testing, Game Testing, Enterprise Application Testing, Web Application Testing, Others
MindTree	Krishnakumar Natarajan	Bangalore	Infrastructure Management & Technical Support, Mobilty, Testing, Data & Analytic Solutions
Nous Infosystems Pvt. Ltd.	Ajith J. Pillai	Bangalore	Unit, Integration, User Acceptance, Functional, Localization and internationalization, Product Migration, Product Security, GUI Testing Performance, Post-testing support, Product documentation
Oak Systems	Sudher	Bangalore	Functional, Integration, Regression, Structural, Component and Usability, Test case design & Test suite preparation, Performance, Load, Scalability and Platform Compatibility
QA Info Tech	Mukesh Sharma	Noida	Accessibility, Software Quality Assurance, Functional, Performance, Security, Globalization, Localization, Cloud, Smart Devices
QA Zone	Smita Mishra	New Delhi	verification, validation & non functional, test automation, test consulting, Test Process Assessment, Metrics & Measurement set up, Data Management, COE Setup
Qutesys Technologies	Rajendra Menon	Bangalore	Test Process Outsourcing, Test Advisory Services, Testing As A

Pvt. Ltd			Service, Certifications and Trust Seals.
S T C Thirdeye Technology	Sriram	Chennai	Requirements Capture and Analysis, Test Planning, Test case Design & Traceability Matrix, Test Execution, Defect Tracking and Management, Test Metrics
Span Systems	Chakravarthy M. Venkataramana	Bangalore	Software Testing, Quality Assurance, Offshoring
Stag Software	T Ashok	Bangalore	Scientific bug hunting activity, Hypothesis Based Testing (HBT)
Thinksoft Global	Asvini Kumar	Chennai	Cards & Payments, Banking, Insurance, Treasury and Capital markets, Financial Product, Test Consulting, Testing Mobile Applications, Test Technology and Tools
Verifaya	Girish N. Basidoni	Bangalore	Automated software testing solutions
ZenQA	Murali Mohan	Hyderabad	Core Functional, Test Automation, API/Web Services/WCF Functional, Performance, Security, Pre-Certification, Globalization or Localization, Mobile application
Zensaar	Ganesh Natrajan	Pune	Verification, Validation services and non functional testing, test automation, specialized testing and test consulting

Milind Limaye on *beyondtesting* highlighted some of the skills that every tester is expected to possess.

5 Skills Every Tester Should Have

1. Communication

Testers are not only expected to be good listeners; however they are also expected to be good presenters as well. Testers need to communicate with the management, the users and the developers before, during and after development, prepare test cases, test logs and present the test reports. Communication skills of a tester include his/her body language, the tone, their writing style as well as the words they use.

2. Domain Knowledge

Although testers are not expected to be domain experts; however they are expected to have a brief understanding about the application. This will help them identify the possible defects a user might face. According to Millind, the tester should keep the domain in mind when deciding on the priority of the bugs and defects, the test cases and the priority of the requirements. They should also be aware of the various domain complexities and the challenges.

3. Desire to Learn

Testers are expected to keep themselves up to date with the various technologies, approaches, tools and techniques and apply them during testing. Testers should always remember that new tools may offer then some new and exciting features which can enhance their testing capabilities.

4. Differentiate the Defects

Testers should have the ability to identify and differentiate the defects which need immediate attention and those that are severe. The test plan should include the various levels with regards to the priorities and severities of the bugs.

5. Planning

Testers must be able to plan the testing process accordingly. The test plan should include the priorities of the various test cases, the number of defects that they are targeting, as well as all the functionalities, requirements and features. A well planned test will lead to a high customer satisfaction.