A Comparison Between Present and Future Models Of Software Engineering

Harikrishnan Natarajan[1], Ram Kumar Somasundaram[2] and Kalpana Lakshmi[3]

[1] Associate Consultant, Testing Practice, Capgemini India, Tamilnadu, India

[2] Associate Consultant, Testing Practice, Capgemini India, Tamilnadu, India

[3] Associate Consultant, Testing Practice, Capgemini India, Tamilnadu, India

Abstract

Every day many of us have been introduced to latest Testing life cycle, Tools, webinars, articles and books published by testing vendors. By enabling feature testing with these data's, our research attempts to answer the question "Requirements Engineering, Development and Testing – Skills Needed by Future Tester". It is concerned with the feature testing era that examine the area of Requirements Engineering, Development and Testing through the development models, which are known as software development life cycle. At presents there are different development models namely, waterfall, Spiral, RAD, V Model, W Model, Systems of Systems, Safety critical systems, Prototyping, Extreme programming and Agile Model. These have resulted in testing era asking a new set of questions to testers, like what sort of Skills Needed by Future Tester?, Factors driving these changes include business dynamics, changes in underlying technology, a deeper understanding of quality and user experience, and now we are at an interesting journey on what the future holds for this discipline, moving into a zone of what sort of skills are becoming inevitable for Future Testers community. When there is a future testing model arrives, this paper examines all these future stuffs with a detailed analysis.

Keywords: *RAD (Rapid Application Development), Systems Development Method (SDM).*

1. Introduction

In the last decade the focus of the software testing discipline has undergone radical changes, ranging from developer-driven testing, and independent testing like black box, white box, to a combination of black and white box testing. Journey of software testing started from pure manual to combination of manual and automated testing. One of the most important things about the skill set of a tester is every software project needs a tester with different skill set and changing role's according to the testing model, dedicated to finding errors, and assuring quality. The time and skills spent by each tester is insignificant, compared to the time needed to fix the errors that are found later in the client's live environment. Depends on the skill sets of the testers it may affect the company image or it may promote the business and keep the business alive with excellence. The tester is the person who inevitably knows the software best. Tester is the one who knows business rules behind the reports, screen or what should happen if user clicks a particular button, what SQL procedure will be invoked and executed if a particular window opens. It is impossible to develop quality software without a skilled tester. The tester can be the one who deals with the User Acceptance Testing, customer support and tester's knows the application best, tester's can also write the help documentation and all other stuff that is something that programmers can abhor and try to avoid at any cost. A typical tester who has the combination of breaking the system attitude, logical thinking, domain knowledge and technical skills on operating system, web application, application server, database, scripting and programming languages will be highly sought after (e.g., the LAMP stack—Linux, Apache, MySQL, PHP) these are the skill required to collaborate with development team on tasks such as static code reviews, understanding the system to achieve and improve test coverage, defect trouble shooting etc and it changes dynamically respect to the changes in dynamic business and technology. A change in business and technology will brings the need of new approach for a new architecture, currently black box lifecycle processes and tools cannot cope with the dynamic need for thorough white box testing. For example, we need to test the User Interface (UI) and the other components behind it. Nowadays lifecycle models are agile and iterative. This in turn increases the importance of automated testing skill sets. This approach enables the continuous integration

testing and it is impossible to work on manual testing and classical tools.

Every project can be classified into four distinct phases - Requirement management, Analysis and Design, Development, Testing and Deployment & Production Management. All these phases are managed by dedicated teams and the focus areas for testers are Requirement, Development and Testing phase. Now here, we are going to closely focus on Requirement, Development, Testing phase, Testing Process, Skill Stacks and the modern approach to future testing. Lets us have a quick look at some of the common challenges we have come across in various models, while we are expecting a future testing, over the recent past.

2. Modern Challenges in Historical

1. Testing processes that span across different systems create highly complex test data.
2. Heterogeneous technologies - Expensive to do testing and managing a wide variety of systems.
3. Managing wide range of Peoples across the geographies in disparate teams.
4. More reusability - More problems in security and performance area.
5. Integrated Systems - More points of failure results in more defects.
6. Multi layered and Mission complex architecture makes it tough to isolate defects.
7. Agility - Enabling faster change leads to increase in integration and regression testing .

3. Initial Steps of Testing Process

Every Software Project has 3 important phases - Requirement, Development and Testing Phases. Software testing Focus on three phases

Requirement Phase: The first step in the software testing is validation of requirements.

Development Phase: We validate the development process (this is synonymous with unit testing). This validation of development process results in standard software development

Testing Phase: The subsequent level of testing phase is all about with system testing, integration testing and regression testing to shield them from breaking changes.

4. Different Models of Testing Life Cycle - Changing Skills & Roles of Testers

Waterfall Model

The waterfall model is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) [2] through the phases of Conception, Initiation, Analysis, Design (validation), Construction, Testing and maintenance [4].

Spiral Model

The spiral model is a software development process combining elements of both design and prototyping-in-stages, in an effort to combine advantages of top-down and bottom-up concepts. [8] Also known as the spiral lifecycle model, it is a systems development method (SDM) used in information technology (IT) [1]. This model of development combines the features of the prototyping model and the waterfall model.

RAD Model

RAD represents 'Rapid Application Development'. In order to implement a RAD development, all of the requirements must be known in advance. But with RAD the requirements are formally documented. Each requirement is categorized into individual components. Then each component is developed and tested in parallel. All this is done in a set period of time. RAD is considered to be an iterative development model.

V Model

The V-Model is an industry standard framework that shows clearly the software development lifecycle in relation to testing. [7] It also highlights the fact that the testing is just as important as the software development itself. The relationships between development and testing are clearly defined. The V-Model improves the presence of the testing activities to display a more balanced approach [3].

W Model

The W-model further clarifies the priority of tasks and dependence between the development and testing activities. Though as simple as V-model, the W-model makes the importance of testing and ordering of the individual testing activities clear. It also clarifies that testing and debugging are not the same thing.

Systems of Systems

A system of systems is a set of collaborating components (including hardware, individual software applications and communications), interconnected to achieve a common purpose, without a unique management structure.

Safety critical systems

Safety critical systems are those which, if their operation is lost or degraded (e.g. as a result of incorrect or inadvertent operation), can result in catastrophic or critical consequences. The supplier of the safety critical system may be liable for damage or compensation, and testing activities are thus used to reduce that liability. The testing activities provide evidence that the system was adequately tested to avoid catastrophic or critical consequences.

Agile Model

Agile Software Development is a conceptual framework for software development that promotes development iterations throughout the life-cycle of the project. Many different types of agile development methods exist today, but most aim to minimize risk by developing software in short amounts of time. Each period of time is referred to as an iteration, which typically lasts from one to four weeks. System was adequately tested to avoid catastrophic or critical consequences.

Future Model

Requirements will be more iterative passion, development will be more iterative and more agility. Hence testing will be more iterative and more agility. So the future testers must have the knowledge of modern tools and methodologies. The roles of tester's are increasingly important and crucial throughout the entire lifecycle model.

5. Modern Tools and Methodology Changes

Modern Tool Requirements

Modern tools has a unique architectural design, they can address testing in non- components, cope up with subscriptions to brokers; can interpret messages that flow across the complex system. Modern tools have its unique protocols such a as SOAP, WS - Security and takes you to have web of component invocations and defect isolation, allows you to have across technologies, example: UI automation, middleware tests, DB querying and service testing.

Methodology Changes

A new future model brings the need for a new approach. Currently with historical lifecycle processes and tools cannot cope with sudden need for deep testing. Tools like HP's automation tools, IBM's tools and many other open source tools pave a way to overcome the challenges needs. But in future testing, depending upon the methodology adapts to diverse system ecology and virtually strong methodologies need to addresses the governance ease of roadblocks implementation. Finally a good future testing tools and modern tools will aid in testing service and also help in defect isolation.

6. Discussion and Conclusion

The Future Tester's has to improve his knowledge in new generation of testing models, approaches and specialized modern testing tools. Future Testers need to know modern testing tools, larger testing processes, skill stacks and wider range of testing roles & practices. These includes Testing Models, Business changes, Technology trends, Test Data Management, Automation, Security, Performance and deep understanding of quality. Success in Future testing will come to Quality Assurance organizations that leverage the testers who have the ability in selecting the right tools and supporting testing process. This move will ensure that the future tester will deliver on the promises that meet the business user's expectation. Such a tester will shine in future Requirements Engineering, Development and Testing.

References

[1]. Ian Sommerville, "Software Engineering", Addison Wesley, 7th edition, 2004.
[2]. CTG. MFA – 003, "A Survey of System Development Process Models", Models for Action Project: Developing Practical Approaches to Electronic Records Management and Preservation, Center for Technology in Government University at Albany / Suny, 1998.
[3]. Steve Easterbrook, "Software Lifecycles", University of Toronto Department of Computer Science, 2001.
[4]. National Instruments Corporation, "Lifecycle Models", 2006, http://zone.ni.com.
[5]. JJ Kuhl, "Project Lifecycle Models: How They Differ and When to Use them", 2002 www.businessesolutions.com.
[6]. Karlm, "Software Lifecycle Models', KTH, 2006.

[7]. Rlewallen, "Software Development Life Cycle Models", 2005, http://codebeter.com.
[8]. Barry Boehm, "Spiral Development: Experience, Principles, and Refinements", edited by Wilfred J. Hansen, 2000
[9]. A. comparison between five Models of Software Engineering, September, 2010

Harikrishnan Natarajan Harikrishnan Natarajan is an Associate Consultant at Capgemini India Pvt. Ltd. He started his career in 2009 and has extended his knowledge and expertise over the years by working in banking areas (Core banking, Retail Banking) and roles (Test Engineer, Test Analyst). He received a B.Tech Degree in Information Technology and he is an ISTQB certified tester. His current research includes Genetic Algorithm, Quality Management, Process Improvement, Software Process Modeling, Software Metrics, and Agile Software Development. He is a member of Association of Computer Electronics and Electrical Engineers (ACEEE).

Ram Kumar Somasundaram is an Associate Consultant at Capgemini India Pvt. Ltd. He completed his B.Tech Information Technology, in Tamilnadu College of Engineering. He has 3 + years of experience in Manual Testing. Also He is an IIBF Certified. He has been involved in the testing Credit Cards, Core banking. He is currently engaged in research involving Testing Maturity Model and Security based testing in Cloud computing.

Kalpana Lakshmi is an Associate Consultant at Capgemini India Pvt. Ltd. she completed her MCA in C R Reddy College, Eluru, Andhrapradesh. She has close to 4 years of experience in Automation and Manual Testing. She has been involved in the testing Banking Products. She is currently engaged in research involving Testing Framework and Quality Management.