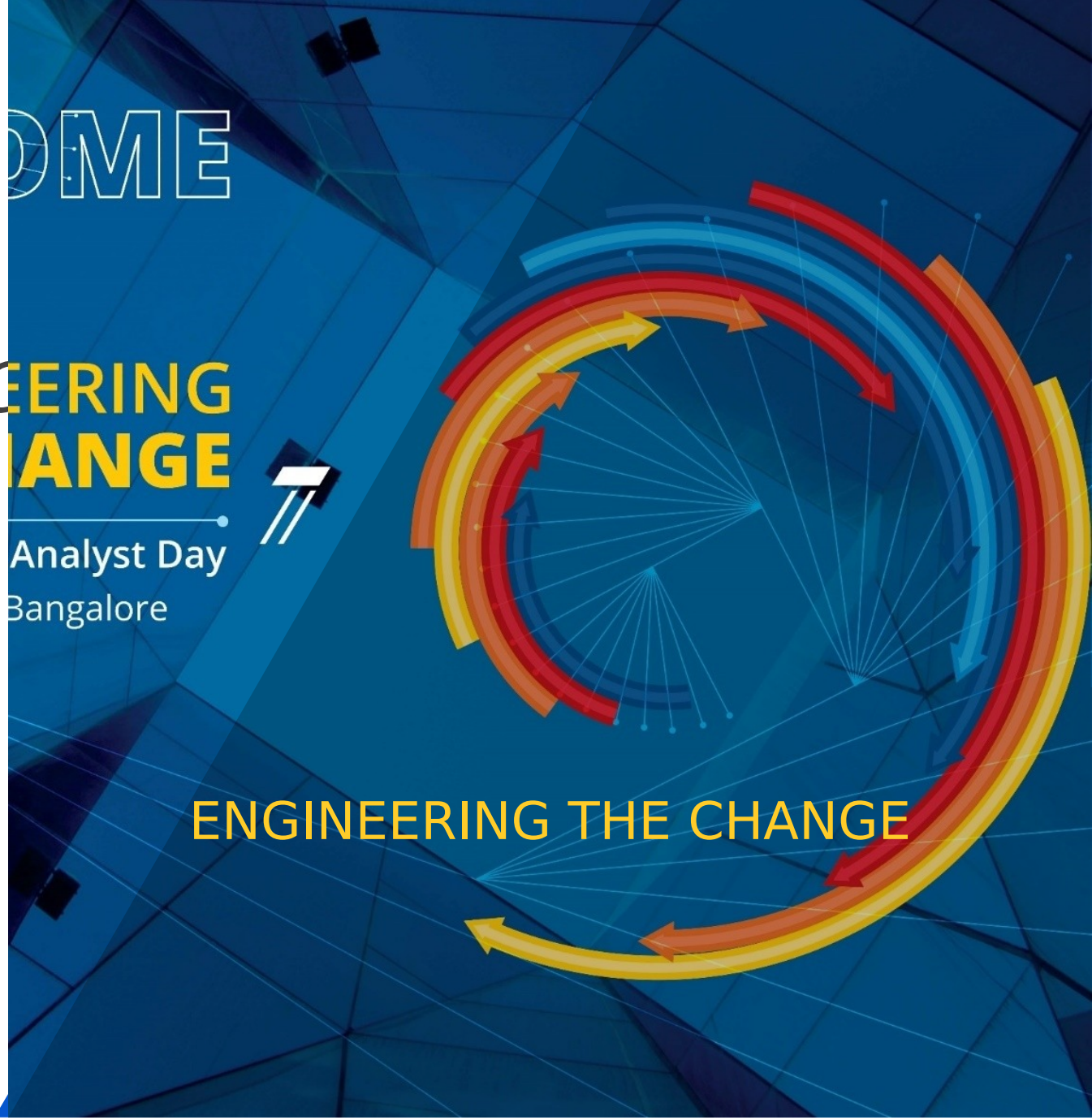
Manual Testing / TDL

L&T Technology Services



Date

Contents

TESTING ON AGILE DEVELOPMENT

What is Agile Methodology

A tester on an Agile project will work differently than one working on a traditional project. Testers must understand the values and principles that underpin Agile projects, and how testers are an integral part of a whole-team approach together with developers and business representatives. The members in an Agile project communicate with each other early and frequently, which helps with removing defects early and developing a quality product.

Agile Software Development and the Agile Manifesto

The Agile Manifesto contains four statements of values:

•Individuals and interactions over processes and tools
•Working software over comprehensive documentation
•Customer collaboration over contract negotiation
•Responding to change over following a plan

The Agile Manifesto argues that although the concepts on the right have value, those on the left have greater value.

Individuals and Interactions

Agile development is very people-centred. Teams of people build software, and it is through continuous communication and interaction, rather than a reliance on tools or processes, that teams can work most effectively.

Working Software

From a customer perspective, working software is much more useful and valuable than overly detailed documentation and it provides an opportunity to give the development team rapid feedback. In addition, because working software, albeit with reduced functionality, is available much earlier in the development lifecycle, Agile development can confer significant time-to-market advantage. Agile development is

What is Agile Methodology (contd.)

Customer Collaboration

Customers often find great difficulty in specifying the system that they require. Collaborating directly with the customer improves the likelihood of understanding exactly what the customer requires. While having contracts with customers may be important, working in regular and close collaboration with them is likely to bring more success to the project.

Responding to Change

Change is inevitable in software projects. The environment in which the business operates, legislation, competitor activity, technology advances, and other factors can have major influences on the project and its objectives. These factors must be accommodated by the development process. As such, having flexibility in work practices to embrace change is more important than simply adhering rigidly to a plan.

Principles

The core Agile Manifesto values are captured in twelve principles:

•Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

•Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

•Deliver working software frequently, at intervals of between a few weeks to a few months, with a preference to the shorter timescale.

•Business people and developers must work together daily throughout the project.

•Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation. Working software is the primary measure of progress.

•Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

•Continuous attention to technical excellence and good design enhances agility.

•Simplicity—the art of maximizing the amount of work not done—is essential.

•The best architectures, requirements, and designs emerge from self-organizing teams.

Agile VS Waterfall Method

What is Waterfall methodology?
Waterfall Model methodology which is also known as Liner Sequential Life Cycle Model. Waterfall Model followed in the sequential order, and so project development team only moves to next phase of development or testing if the previous step completed successfully.

What is the Agile methodology?
Agile methodology is a practice that helps continuous iteration of development and testing in the software development process. In this model, development and testing activities are concurrent, unlike the Waterfall model. This process allows more communication between customers, developers, managers, and testers.

Advantages of Waterfall Model:
•It is one the easiest model to manage. Because of its nature, each phase has specific deliverables and a review process.
•It works well for smaller size projects where requirements are easily understandable.
•Faster delivery of the project
•Process and results are well documented.
•Easily adaptable method for shifting teams
•This project management methodology is beneficial to manage dependencies.

Advantages of the Agile Model:
•It is focused client process. So, it makes sure that the client is continuously involved during every stage.
•Agile teams are extremely motivated and self-organized so it likely to provide a better result from the development projects.
•Agile software development method assures that quality of the development is maintained
•The process is completely based on the incremental progress. Therefore, the client and team know exactly what is complete and what is not. This reduces risk in the development process.

Agile VS Waterfall Method (contd.)

Limitations of Waterfall Model:
It is not an ideal model for a large size project
If the requirement is not clear at the beginning, it is a less effective method.
Very difficult to move back to makes changes in the previous phases.
The testing process starts once development is over. Hence, it has high chances of bugs to be found later in development where they are expensive to fix.

Limitations of Agile Model
It is not useful method for small development projects.
It requires an expert to take important decisions in the meeting.
Cost of implementing an agile method is little more compared to other development methodologies.
The project can easily go off track if the project manager is not clear what outcome he/she wants.

Agile	Waterfall
It separates the project development lifecycle into sprints.	Software development process is divided into distinct phases.
It follows an incremental approach	Waterfall methodology is a sequential design process.
Agile methodology is known for its flexibility.	Waterfall is a structured software development methodology so most times it can be quite rigid.
Agile can be considered as a collection of many different projects.	Software development will be completed as one single project.
Agile is quite a flexible method which allows changes to be made in the project development requirements	There is no scope of changing the requirements once the project development starts.

Agile VS Waterfall Method (contd.)

Agile methodology, follow an iterative development approach because of this planning, development, prototyping and other software development phases may appear more than once.	All the project development phases like designing, development, testing, etc. are completed once in the Waterfall model.
Test plan is reviewed after each sprint	The test plan is rarely discussed during the test phase.
Agile development is a process in which the requirements are expected to change and evolve.	The method is ideal for projects which have definite requirements and changes not at all expected.
In Agile methodology, testing is performed concurrently with software development.	In this methodology, the "Testing" phase comes after the "Build" phase
Agile introduces a product mindset where the software product satisfies needs of its end customers and changes itself as per the customer's demands.	This model shows a project mindset and places its focus completely on accomplishing the project.
Agile methdology works exceptionally well with Time & Materials or non-fixed funding. It may increase stress in fixed-price scenarios.	Reduces risk in the firm fixed price contracts by getting risk agreement at the beginning of the process.
Prefers small but dedicated teams with a high degree of coordination and synchronization.	Team coordination/synchronization is very limited.
Products owner with team prepares requirements just about every day during a project.	Business analyst prepares requirements before the beginning of the project.
Test team can take part in the requirements change without problems.	It is difficult for the test to initiate any change in requirements.
Description of project details can be altered anytime during the SDLC process.	Detail description needs to implement waterfall software development approach.
The Agile Team members are interchangeable, as a result, they work faster. There is also no need for project managers because the projects are managed by the entire team	In the waterfall method, the process is always straightforward so, project manager plays an essential role during every stage of SDLC.

Agile VS Waterfall Method (contd.)

KEY DIFFERENCE

- Waterfall is a Liner Sequential Life Cycle Model whereas Agile is a continuous iteration of development and testing in the software development process.
- Agile methodology is known for its flexibility whereas Waterfall is a structured software development methodology.
- Agile follows an incremental approach whereas the Waterfall methodology is a sequential design process.
- Agile performs testing concurrently with software development whereas in Waterfall methodology testing comes after the “Build” phase.
- Agile allows changes in project development requirement whereas Waterfall has no scope of changing the requirements once the project development starts.

Overview of Agile Testing/TDD/BDD/Automation Approach

There are certain testing practices that can be followed in every development project (agile or not) to produce quality products. These include writing tests in advance to express proper behavior, focusing on early defect prevention, detection, and removal, and ensuring that the right test types are run at the right time and as part of the right test level. Agile practitioners aim to introduce these practices early. Testers in Agile projects play a key role in guiding the use of these testing practices throughout the lifecycle.

Test-Driven Development

Test-driven development(TDD)is used to develop code guided by automated test cases. The process for test-driven development is:

•Add a test that captures the programmer's concept of the desired functioning of a small piece of code
•Run the test, which should fail since the code doesn't exist
•Write the code and run the test in a tight loop until the test passes
•Refactor the code after the test is passed, re-running the test to ensure it continues to pass against the re-factored code
•Repeat this process for the next small piece of code, running the previous tests as well as the added tests

Benefits of Test-Driven Development:

•Helps reduce the amount of time required for rework
•Helps explore bugs or errors very quickly
•Helps get faster feedback
•Encourages the development of cleaner and better designs
•Enhances the productivity of the programmer
•Allows any team member to start working on the code in the absence of a specific team member. This encourages knowledge sharing and collaboration
•Gives the programmer confidence to change the large architecture of an application easily

Overview of Agile Testing/TDD/BDD/Automation Approach (contd.)

Behavioural-Driven Development

Business-Driven Development (BDD) is a testing approach derived from the Test-Driven Development (TDD) methodology. In BDD, tests are mainly based on systems behaviour. This approach defines various ways to develop a feature based on its behaviour. In most cases, the ***Given-When-Then*** approach is used for writing test cases. Let's take an example for better understanding:

- **Given** the user has entered valid login credentials
- **When** a user clicks on the login button
- **Then** display the successful validation message

Key benefits of Behavioural-Driven Development approach:

- Helps reach a wider audience by the usage of non-technical language
- Focuses on how the system should behave from the customer's and the developer's perspective
- BDD Is a cost-effective technique
- Reduces efforts needed to verify any post-deployment defects

How does BDD help in SDLC?

Debugging the errors in the latter stages of the development life cycle often proves to be very expensive. In the majority of cases, ambiguity in understanding the requirements is the root cause behind this. One needs to ensure that all the development efforts remain aligned towards fulfilling pre-determined requirements. BDD allows developers to do the above by :

- Allowing the requirements to be defined in a standard approach using simple English
- Providing several ways to illustrate real-world scenarios for understanding requirements
- Providing a platform that enables the tech and non-tech teams to collaborate and understand the requirements

Scrum

Scrum is an agile framework for developing, delivering, and sustaining complex products, with an initial emphasis on software development, although it has been used in other fields including research, sales, marketing and advanced technologies. It is designed for teams of ten or fewer members, who break their work into goals that can be completed within time-boxed iterations, called *sprints*, no longer than one month and most commonly two weeks. The Scrum Team track progress in 15-minute time-boxed daily meetings, called daily scrums. At the end of the sprint, the team holds sprint review, to demonstrate the work done, and sprint retrospective to continuously improve.

Scrum relies on cross-functional teams to deliver products and services in ***short cycles***, enabling:

- Fast feedback
- Continuous improvement
- Rapid adaptation to change
- Accelerated delivery

Understanding the Scrum Flow

At its heart, Scrum works by breaking large products and services into small pieces that can be completed (and **potentially released**) by a cross-functional team in a short timeframe. Scrum teams **inspect** each batch of functionality as it is completed and then adapt what will be created next based on **learning** and **feedback**, **minimizing risk and reducing waste**. This cycle repeats until the full product or service is delivered—one that meets customer needs because the business has the opportunity to adjust the fit at the end of each timeframe.

Benefits of Using Scrum

Because Scrum teams are continuously creating only the highest priority chunks of functionality, the business is assured maximum return on investment. Scrum helps businesses:

- Innovate faster
- Move from idea to delivery more quickly
- Drive higher customer satisfaction

Role of Tester in Agile

The role of a tester in an Agile team includes activities that generate and provide feedback not only on test status, test progress, and product quality, but also on process quality. In addition to the activities described else where in this syllabus, these activities include:

- Understanding, implementing, and updating the test strategy
- Measuring and reporting test coverage across all applicable coverage dimensions
- Ensuring proper use of testing tools
- Configuring, using, and managing test environments and test data
- Reporting defects and working with the team to resolve them
- Coaching other team members in relevant aspects of testing
- Ensuring the appropriate testing tasks are scheduled during release and iteration planning
- Actively collaborating with developers and business stakeholders to clarify requirements, especially in terms of testability, consistency, and completeness
- Participating proactively in team retrospectives, suggesting and implementing improvements.

Within an Agile team, each team member is responsible for product quality and plays a role in performing test-related tasks. Agile organizations may encounter some test-related organizational risks:

- Testers work so closely to developers that they lose the appropriate tester mindset
- Testers become tolerant of or silent about inefficient, ineffective, or low-quality practices within the team
- Testers cannot keep pace with the incoming changes in time-constrained iterations

Agile Software Development Approaches

Extreme Programming

Extreme Programming (XP),originally introduced by Kent Beck, is an Agile approach to software development described by certain values, principles, and development practices. XP embraces five values to guide development: communication, simplicity, feedback, courage, and respect. XP describes a set of principles as additional guidelines: humanity, economics, mutual benefit, self-similarity, improvement, diversity, reflection, flow, opportunity, redundancy, failure, quality, baby steps, and accepted responsibility.

Kanban

Kanban is a management approach that is sometimes used in Agile projects. The general objective is to visualize and optimize the flow of work within a value-added chain. Kanban utilizes three instruments:

- **Kanban Board**: The value chain to be managed is visualized by a Kanban board. Each column shows a station, which is a set of related activities, e.g., development or testing. The items to be produced or tasks to be processed are symbolized by tickets moving from left to right across the board through the stations.
- **Work-in-Progress Limit**: The amount of parallel active tasks is strictly limited. This is controlled by the maximum number of tickets allowed for a station and/or globally for the board. Whenever a station has free capacity, the worker pulls a ticket from the predecessor station.
- **Lead Time**: Kanban is used to optimize the continuous flow of tasks by minimizing the (average) lead time for the complete value stream. Kanban features some similarities to Scrum. In both frameworks, visualizing the active tasks (e.g., on a public whiteboard) provides transparency of content and progress of tasks. Tasks not yet scheduled are waiting in a backlog and moved onto the Kanban board as soon as there is new space (production capacity) available. Iterations or sprints are optional in Kanban. The Kanban process allows releasing its deliverables item by item, rather than as part of a release. Timeboxing as a synchronizing mechanism, therefore, is optional, unlike in Scrum, which synchronizes all tasks within a sprint.

THANK YOU