ARCHITECTING PROJECT MANAGEMENT

for Enterprise Agility…

July 14 to 16, 2016,

NIMHANS Convention Centre, Bengaluru

Enable Organization with Agile using Tooling/Technology

Leverage of Technology

Paper Id: PMIBC_16_3_004

Author: Mr. Mirza Farhatullah Baig

CONTENTS

ABSTRACT

In Traditional methodology, It is difficult to go back to the eminent changes. Customer is informed only at the end of the Delivery in Tradition methodology, continuous planning for project is the biggest problem. Finding bugs at the end of implementation is blending then we write code again from starting. Agile is the solution to the above listed problems with traditional methodology. Tooling plays a major role in enabling the organization towards Agile. Agile needs multiple steps to complete a project as product backlog, sprint planning, spring backlog, daily scrum, test cases, results, sprint review meeting, release burn chart. Application Life cycle Management tools provide Sprint planning, Daily Scrum meeting, Bidirectional Traceability, Burn-down Charts, Kanban Dashboards. With Support of the Tooling, the Organization Workforce can have a very less manual effort and maintainability. Agile can be easily enabled into the Organization using Tooling and Technologies. Multiple Case Studies have shown that Agile Methodology along with proper Tooling/Technology gives an Organization Edge over the Market in this VUCA (Volatile, Uncertainty, Complexity, and Ambiguity) world.

INTRODUCTION

Agile means "Able to move quickly and easily". Agile is related to a method of project management mainly for software development, by division of tasks into short phases of work and frequent reassessment of plans. The Project Management started using agile methodology to introduce incremental software development model. Scrum is an iterative and incremental agile software development framework for managing product development. It defines "a flexible, holistic product development strategy where a development team works as a unit to reach a common goal". Tooling can play a vital role in achieving the incremental software development. Tooling can provide complete overview of the software development in form of Report. The Scrum Master can maintain the regular check on the Kanban Board. Product Owner will write customer centric items typically stories, rank them and prioritize them directly in the tool. The tool can also be use a communication model between stakeholder and development team. The Sprint planning and Product backlog can be prepared in the tool. The whole Scrum Framework can be built up using an Application Lifecycle Management

DETAILS OF THE PAPER

What is Agile Methodology?

Agile software development life cycle model is a combination of iterative and incremental process models with focus on

Process adaptability and customer satisfaction by rapid delivery of working software product.

Agile Model

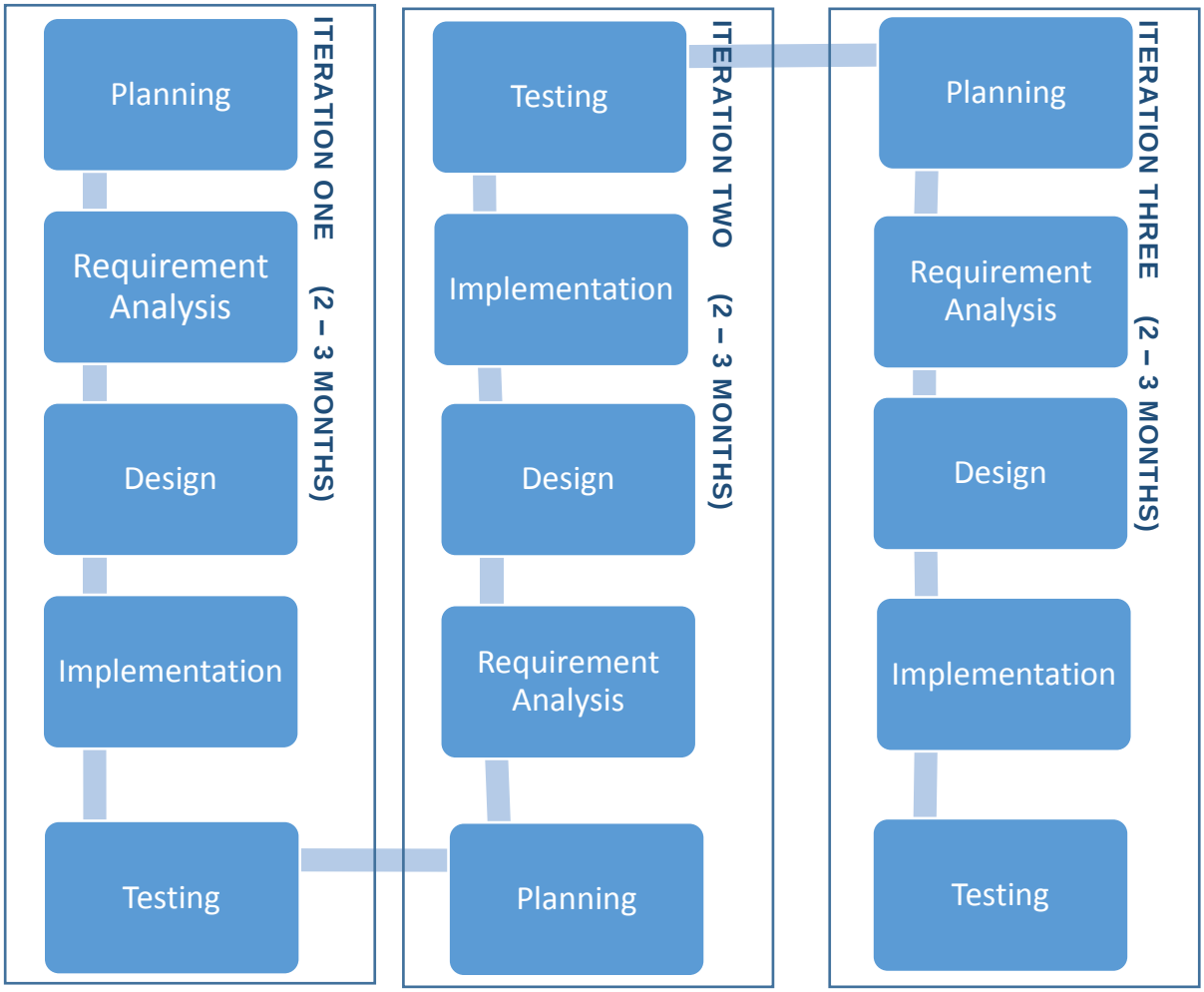


Fig 1: Agile Working Structure

How is Agile followed?

- Agile Methods break the product into small incremental builds.
- These builds are provided in iterations.
- Each iteration typically lasts from about one to three weeks.
- Every iteration involves cross functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing, and acceptance testing.

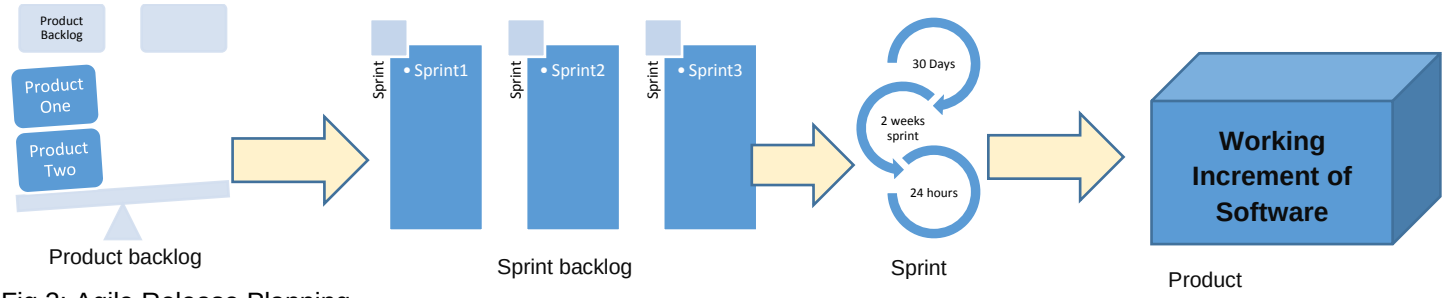


Fig 2: Agile Release Planning

Why Agile should be used?

- Agile is based on the adaptive software development methods where there is no detailed planning and there is clarity on future tasks only in respect of what features need to be developed.
- There is feature driven development and the team adapts to the changing product requirements dynamically.
- The product is tested very frequently, through the release iterations, minimizing the risk of any major failures in future.
- There is no compromise to Quality
- It produces to agreed and fixed Time and Cost

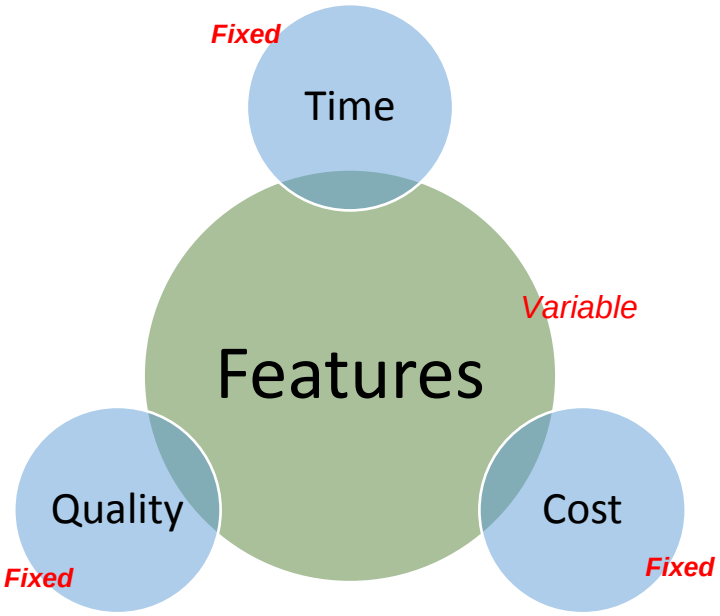


Fig 3: Agile Background

Principles of Agile:

- **Individuals and interactions** - in agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.
- **Working software** - Demo working software is considered the best means of communication with the customer to understand their requirement, instead of just depending on documentation.
- **Customer collaboration** - As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.
- **Responding to change** - agile development is focused on quick responses to change and continuous development.

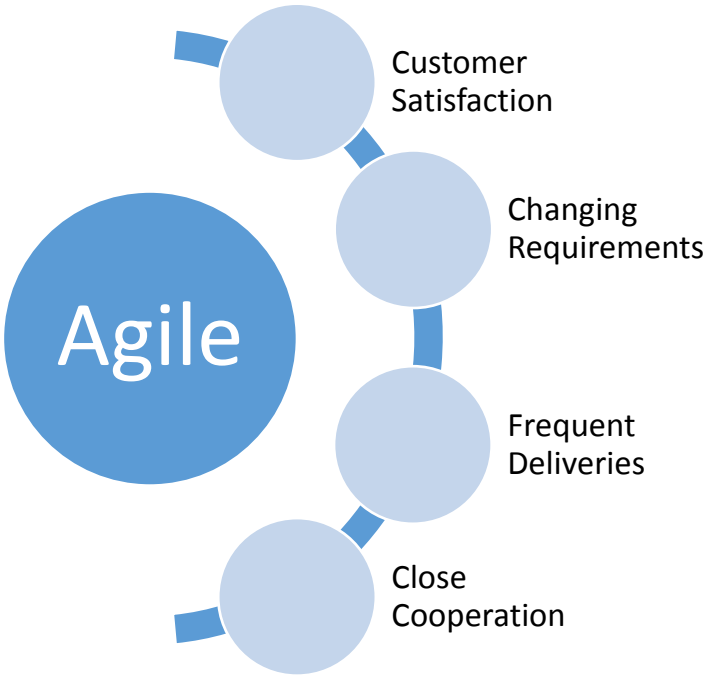


Fig 4: Agile Basic Principles

Advantages:

- Promotes teamwork and cross training.
- Functionality can be developed rapidly and demonstrated.
- Resource requirements are minimum.
- Suitable for fixed or changing requirements.
- Delivers early partial working solutions.
- Good model for environments that change steadily.
- Minimal rules, documentation easily employed.
- Enables concurrent development and delivery within an overall planned context.
- Little or no planning required, easy to manage, gives flexibility to developers.

Disadvantages:

- Not suitable for handling complex dependencies.
- More risk of sustainability, maintainability and extensibility.

- An overall plan, an agile leader and agile PM practice is a must without which it will not work.
- Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.
- Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
- There is very high individual dependency, since there is minimum documentation generated.
- Transfer of technology to new team members may be quite challenging due to lack of documentation.

Agile vs Traditional SDLC Models:

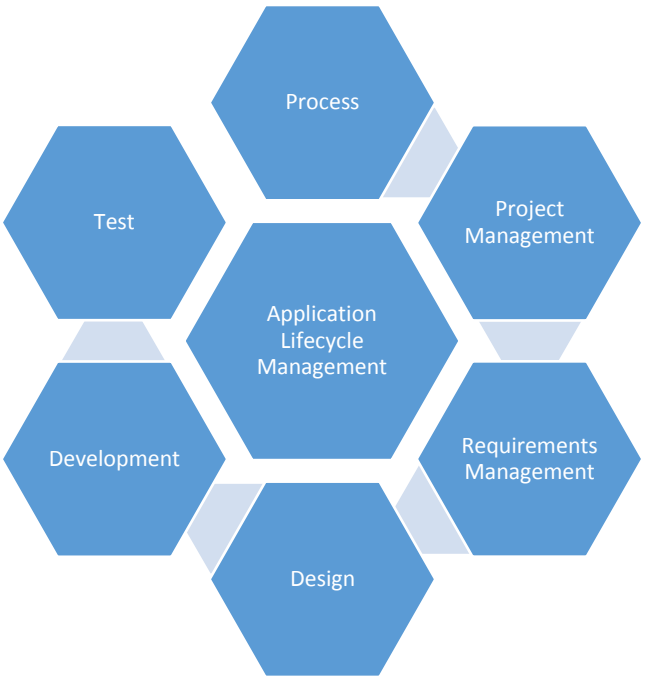
- Agile is based on the adaptive software development methods whereas the traditional software development lifecycle models like waterfall model is based on predictive approach.
- Predictive teams in the traditional software development lifecycle models usually work with detailed planning and have a complete forecast of the exact tasks and features to be delivered in the next few months or during the product life cycle. Predictive methods entirely depend on the requirement analysis and planning done in the beginning of cycle. Any changes to be incorporated go through a strict change control management and prioritization.

S No	Agile	Traditional
1	Incremental Value & Risk Management	Phased approach with an attempt to know everything at start
2	Embracing Change	Change Prevention
3	Delivery Early	Delivers value at the end
4	Transparency	Detailed planning
5	Inspect and Adapt	Meta Solution
6	Self-Managed	Command and Control
7	Continual Learning	Learning is secondary

Table 1: Agile vs Traditional methods

What is ALM?

Application **L**ifecycle **M**anagement refers to the capability to integrate, coordinate and manage the different phases of the software delivery process. From development to deployment, ALM is a set of pre-defined process and tools that include definition, design, development, testing, deployment and management. Throughout the ALM process, each of these steps are closely monitored and controlled.



Importance of ALM:

Management of business assets requires knowing what assets you have, their usage patterns and owners, and whether they are satisfying business needs.

Cost reduction frees up capital and funding for improvements and innovations.

Fig 5: ALM Base Structure

Why ALM :

- Provides a repository for all testing assets and provides a clear foundation for the entire testing process
- Establishes seamless integration and smooth information flow from one stage of the testing process to the next
- Supports the analysis of test data and coverage statistics, and provides a clear picture of the accuracy and quality of an application at each point in the lifecycle
- Provides a consistent repeatable process for:
    - Capturing test requirements
    - Planning, developing, scheduling, and executing tests
    - Analyzing test results
    - Managing defects and issues

Agile ALM:

In addition to the benefits described above, the ALM system can be indispensable in implementing Agile or Lean processes. For teams that are working in Agile or Lean environments, the ALM system can provide the infrastructure for collecting metrics on velocity, flow, and queuing and cycle time. These are necessary measures to truly deliver on the value promised by agile methodologies.

In a Scrum environment, the ALM system can track changes as they are completed and generate Sprint or Release burndown (or burnup) reports. At the end of each Sprint it can track the number of story points delivered by each team. Over time, this will allow the teams to establish their delivery 'velocity'. That velocity becomes a critical component in creating predictable estimates of future deliveries.

In Kanban or other Lean environments, the ALM system can be used to establish limits on work-in-progress — improving the flow of changes. The time that changes spend in queues can be measured and improved upon. This will allow for the removal of bottlenecks and improvements in cycle time. Over time, predictable cycle times for changes can be established.

Agile ALM is a guide for Java developers who want to integrate flexible agile practices and lightweight tooling along all phases of the software development process. The book introduces a new vision for managing change in requirements and process more efficiently and flexibly. It synthesizes technical and functional elements to provide a comprehensive approach to software development.

Agile Application Lifecycle Management (Agile ALM) combines flexible processes with lightweight tools in a comprehensive and practical approach to building, testing, integrating, and deploying software. Taking an agile approach to ALM improves product quality, reduces time to market, and makes for happier developers.

CONCLUSION

Agile means "Able to move quickly and easily". Agile is related to a method of project management mainly for software development, by division of tasks into short phases of work and frequent reassessment of plans. Project Management started using agile methodology to introduce incremental software development model.

REFERENCES

www.wikipedia.org

Agile Communities

ALM Pages on Google Search

Image is captured from wikipedia