



Lifecycle and Requirements-II



Manas Jyoti Das, PhD
Computer Science

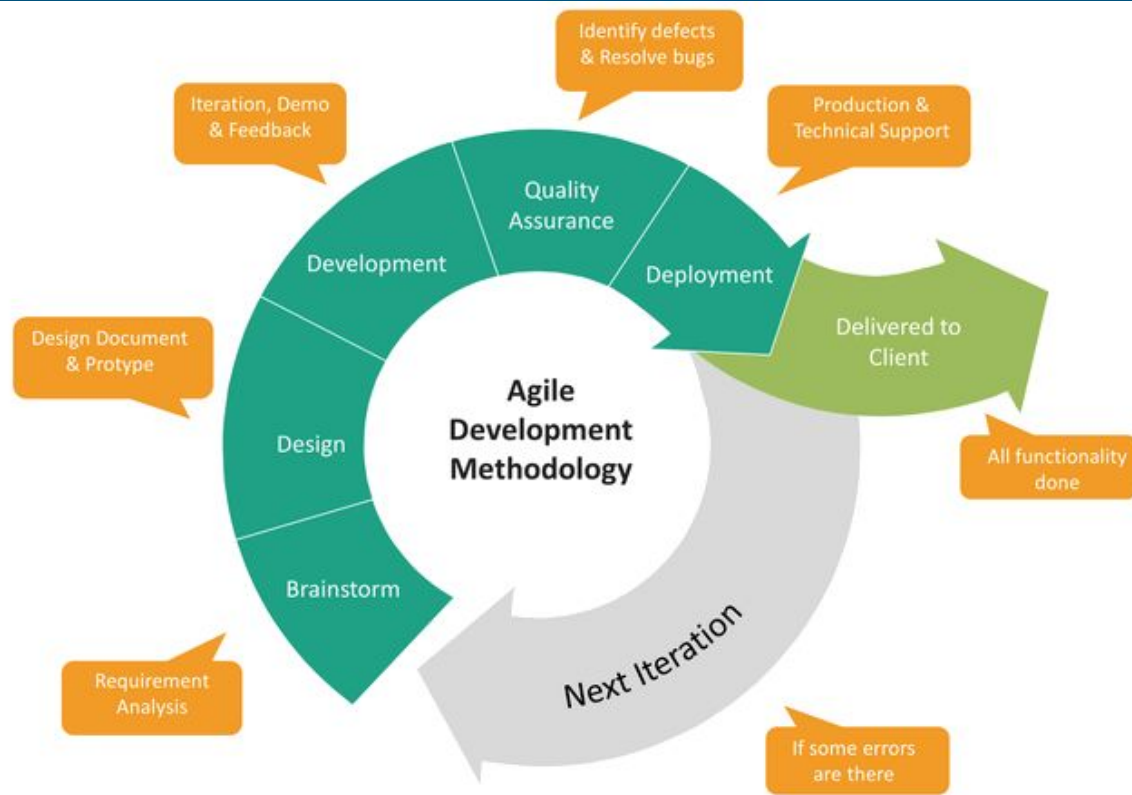


I think therefore I am

- You want to develop software faster, what will you do?
 - How are you going to document the user requirements?
 - Are you going to implement all the features?
 - What is the best way to develop, I mean the team structure?
 - Communication within the team?
 - If you are not going to implement all the features, how are you going to add new features?
What is the duration?
 - Team structure (manager?)
 - Team meeting frequency?
 - Let's create different phases...



Agile



Agile

- Meaning: easily moved, light, active software process
- Developed with the view that projects adapt to change requests
- The requirements are decomposed into many small incremental parts that can be developed over one to four weeks each
- Emphasis on working software over comprehensive documentation
- Customer collaboration over contract negotiation (make customer representative part of the team)
- Responding to changes are welcome over following a rigid plan
- Basically you want to move to the market fast

Agile methodology

- Agile is a umbrella term used for:
 - XP (extreme programming)
 - Scrum
 - Unified process
 - Crystal
 - Lean

Agile model

- Requirement should be in the form of a user story, so it will be informal
- Development can be based on the stories
- Usually each story corresponds to a single functionality or requirement
- These stories if combined together may produce the whole functionality of the software system
- A user story in agile software development is a concise description of a software feature from the perspective of an end user or customer
- The combined user stories can be used to find:
 - How complex the work is and how much work needed to be done
 - Whether the software is feasible and risks involved in it
 - The reliance between different functionalities

Agile model

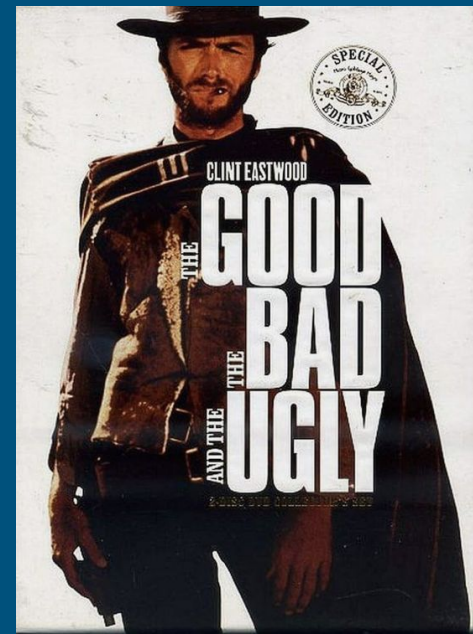
- Face-to-face communication favoured over written documentation
- Team members should communicate with each other rather passing documents to each other
- To facilitate this:
 - Team sizes are usually small
 - Development team shares a single office space
 - This makes the agile model more suitable for small project development
 - Favoured by start-ups

Agile model: principles

- The primary measure of progress:
 - Incremental release of working software
- Important principles:
 - Frequent delivery of versions – once every few weeks
 - Requirement changes are easily accommodated
 - Close cooperation between customer and developers
 - Face-to-face communication among team members

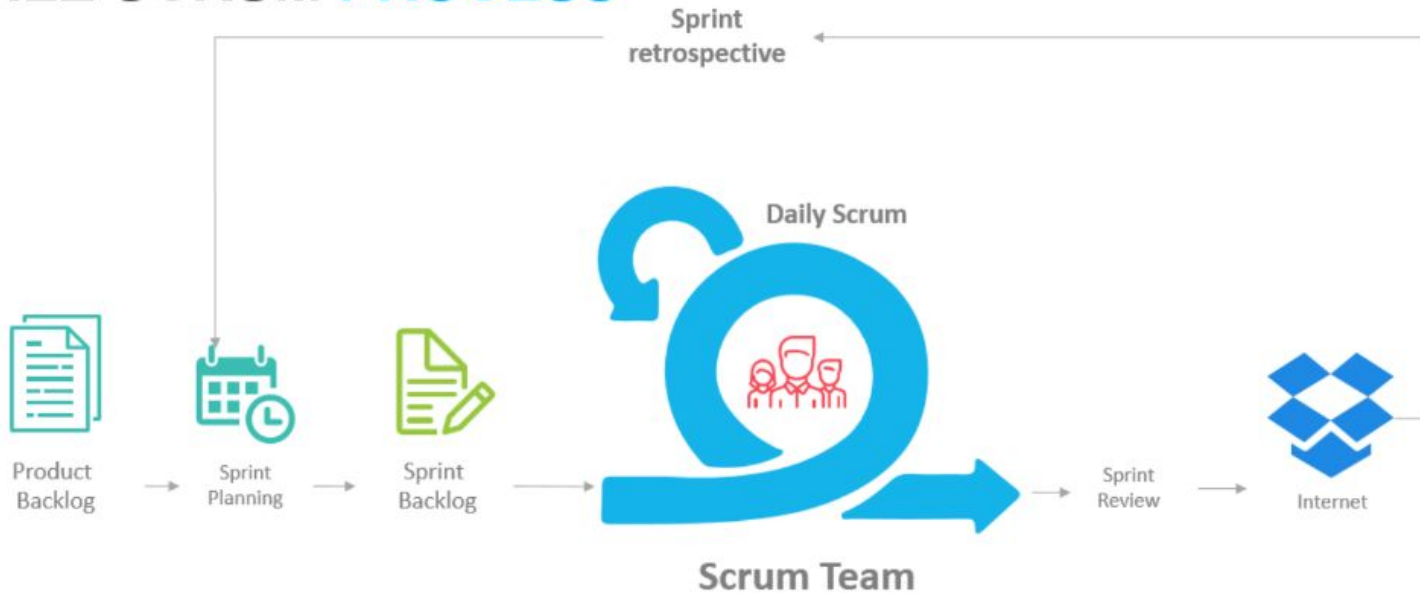
The good, The bad and The ugly

- Good:
 - Produces software in least time
 - Least cost
 - Good quality
 - Accommodate customer requests
- Bad:
 - Several interpretation is possible of a story, so high quality skill people are required
 - No long term plan only focus on one increment
 - Focus on face-to-face communication so knowledge is with the developer, developer leaves the knowledge will vanish
- Ugly:
 - Since customer keeps on getting feedback, feature set may increase
 - Customer expectation may increase
 - Difficult to quantify cost, time and quality
 - After the product is delivered and team disperses (to other project) difficult to maintain

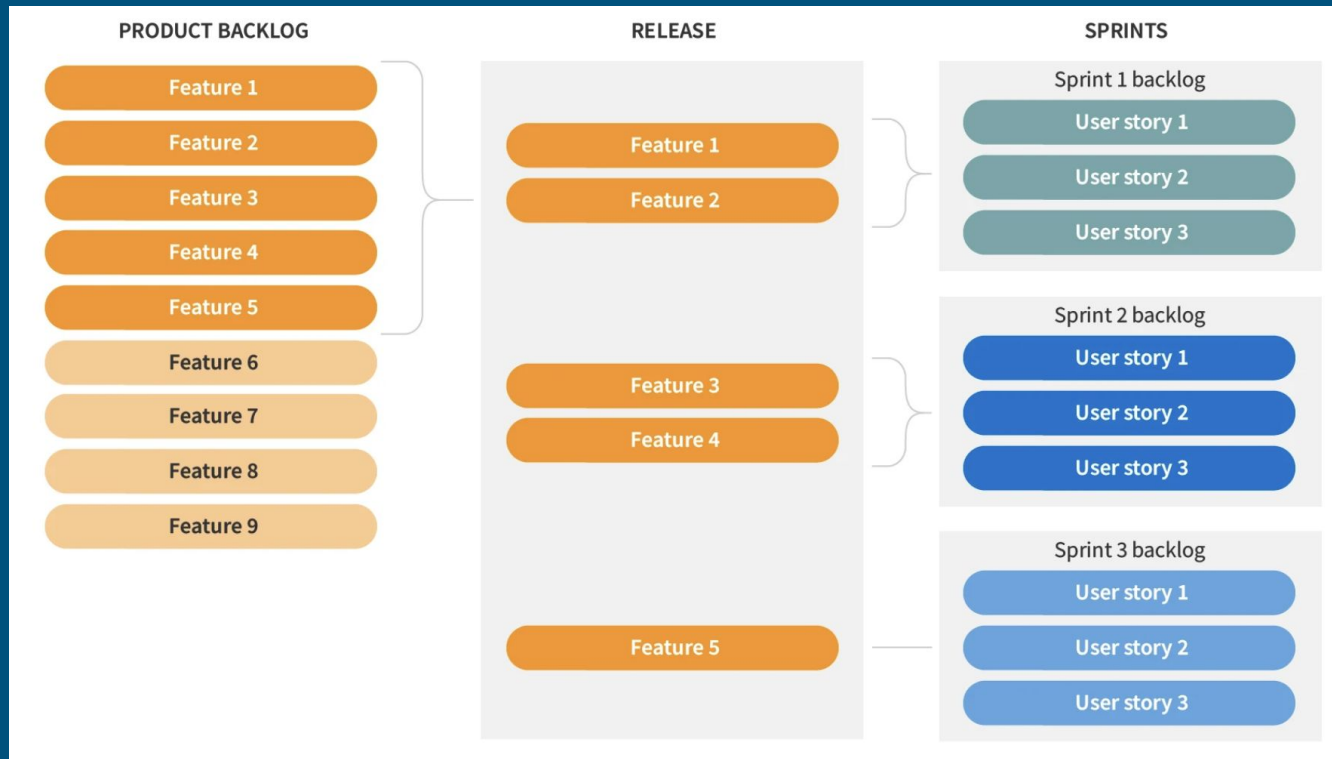


Scrum

AGILE SCRUM PROCESS



Scrum



Scrum

- Increments are now in months, these increments are called sprints
- Software requirement are captured as a list called product backlog
- Product backlogs are user stories
- Scrum progresses in **sprints** usually of one month
- Software increment is designed, coded and tested during the sprint
- No changes are entertained during the sprint

Scrum

- Roles:
 - product owner (a customer or member of the team),
 - scrumMaster(Project manager),
 - team
- Ceremonies: Sprint planning, sprint review, sprint retrospect, and daily scrum meeting
- Artifacts: product backlog, sprint backlog and burndown charts

User stories

- In each iteration the team implement the highest priority user story
- “As a customer, I want to be able to filter products by price, so that I can easily find the products that I am interested in”
- As a <user role>, I want to <goal>, so that <benefit>
- The story outcome should be measurable, and will help the developer to formulate the problem
- Collaborate with the stakeholders to ensure that the user stories are complete and accurate
- Be specific and measurable. What features and functionality are needed to meet the user's needs?

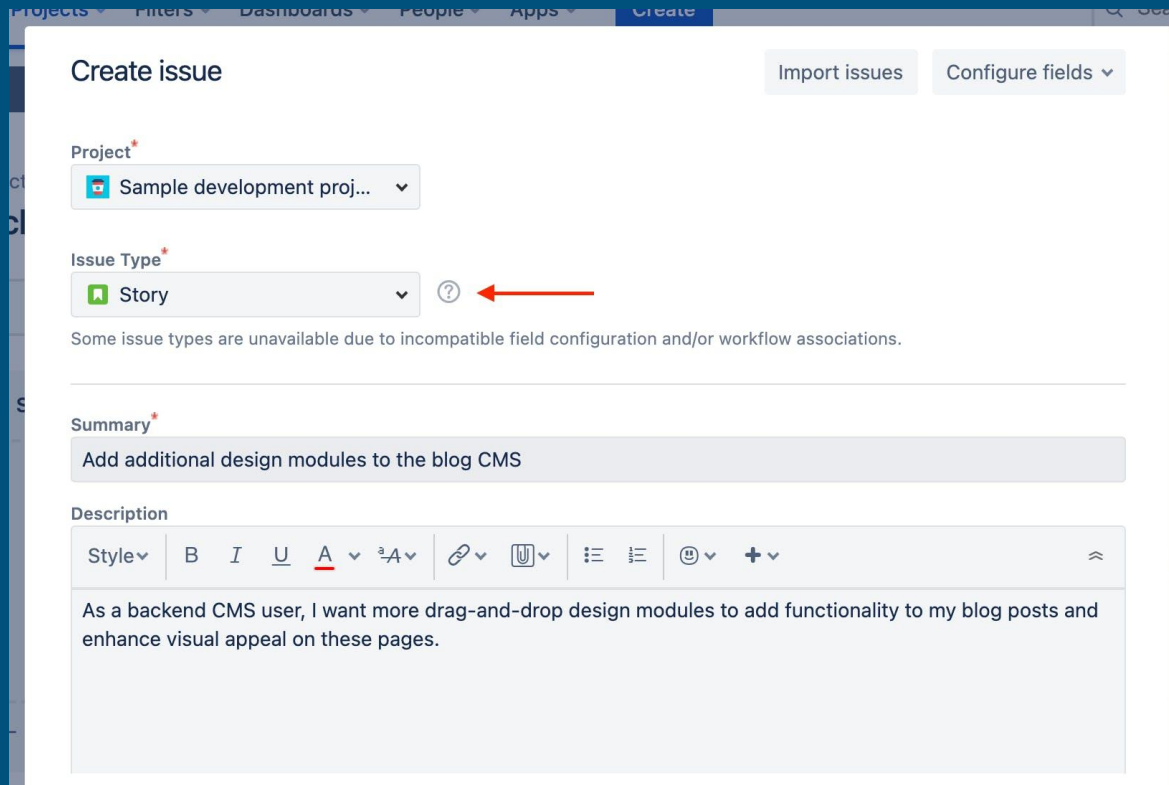
User story, key components

- Each user story consists of three key components:
 - **Role:** It's usually written in the form of "As a [role]"
 - **Goal:** The goal describes what the user wants to achieve or what problem they want to solve using the software. It's written in the form of "I want to [goal]"
 - **Benefit:** This component explains why the user wants to achieve the goal and what value it brings to them or the business. It's written as "So that [benefit]"
- “As a website visitor, I want to be able to create an account on the site so that I can save my preferences and easily access my order history”

User story, few other components


- There are two more components that can be part of the user stories
- **Description:**
 - Provides a high-level overview of the feature or functionality from the user's perspective
 - Can add additional documentation, business logic
- **Acceptance:**
 - Use unit testing: A unit can be a function, method, module, object, or other entity in a software
 - The acceptance criteria of a user story are a set of conditions that must be met in order for the user story to be considered complete
 - Acceptance criteria provide specific and testable details, leaving no room for ambiguity
 - The combination of the user story and its acceptance criteria ensures that the development team understands what needs to be built and how to determine when it's done.

User story



Create issue Import issues Configure fields ▾

Project*
Sample development proj... ▾

Issue Type*
Story ▾ ⓘ 

Some issue types are unavailable due to incompatible field configuration and/or workflow associations.

Summary*
Add additional design modules to the blog CMS

Description

Style ▾ B I U A ▾ A ▾ Link ▾ U ▾ List ▾ List ▾ Emojis ▾ + ▾ ≡

As a backend CMS user, I want more drag-and-drop design modules to add functionality to my blog posts and enhance visual appeal on these pages.

Other user story template

- As a <who> I want <what> so that <why>, description and acceptance criteria
- As a <user type> I want <function> so that <benefit>, description and acceptance criteria

User stories characteristics

- **Independent:** User stories should be independent of each other, so that they can be completed and delivered in any order
- **Negotiable:** User stories should be negotiable, so that the development team and the product owner can collaborate to define the scope and acceptance criteria of the story
- **Valuable:** User stories should deliver value to the user, either by solving a problem or providing a new feature
- **Estimable:** User stories should be estimable, so that the development team can estimate how long they will take to complete
- **Small:** User stories should be small enough to be completed within a single iteration
- **Testable:** User stories should be testable, so that the development team can verify that they have been completed correctly



Yes SMART!!

- **Specific:** The goal should be specific and well-defined. What exactly do you want to achieve?
- **Measurable:** The goal should be measurable, so that you can track your progress and determine when the goal has been achieved.
- **Achievable:** The goal should be achievable, given the available resources and constraints.
- **Relevant:** The goal should be relevant to the overall goals of the team and the business.
- **Time-bound:** The goal should have a specific deadline by which it should be achieved.

- =====
- SMART in agile development refers to a set of criteria that can be used to assess acceptance criteria

Parts of SRS document

- Four important parts of SRS document
- 1. Functional requirement
 - Output for a given inputs and relationship between them
 - Should specify invalid inputs
 - Describe the purpose of the function
 - It should showcase functions that will carry out some meaningful piece of work
- 2. External interfaces
- 3. Non-functional requirements
- 4. Constraints: e.g. open source tools, open data sets, etc.

External Interfaces

- User interfaces
- **Hardware interfaces** : With other hardware drivers, e.g. GPU etc.
- **Software interfaces** : Need other software to run e.g. .NET framework, activeX
- **Communication interfaces** : Type of communication, its speed and other bandwidth
- **File export format**: CSV, XML, JSON, JPEG, MP4 and many more

Non functional requirement

- Maintainability
- Portability
- Usability : GUI (standalone) or browser
- Security
- Safety
- Reliability : If it fails how quickly can it recover
- Performance issue: How fast the software will produce a result
- Many more

Constraints

- Hardware
- Operating system
- I/O devices to be used
- etc.

Thank you
