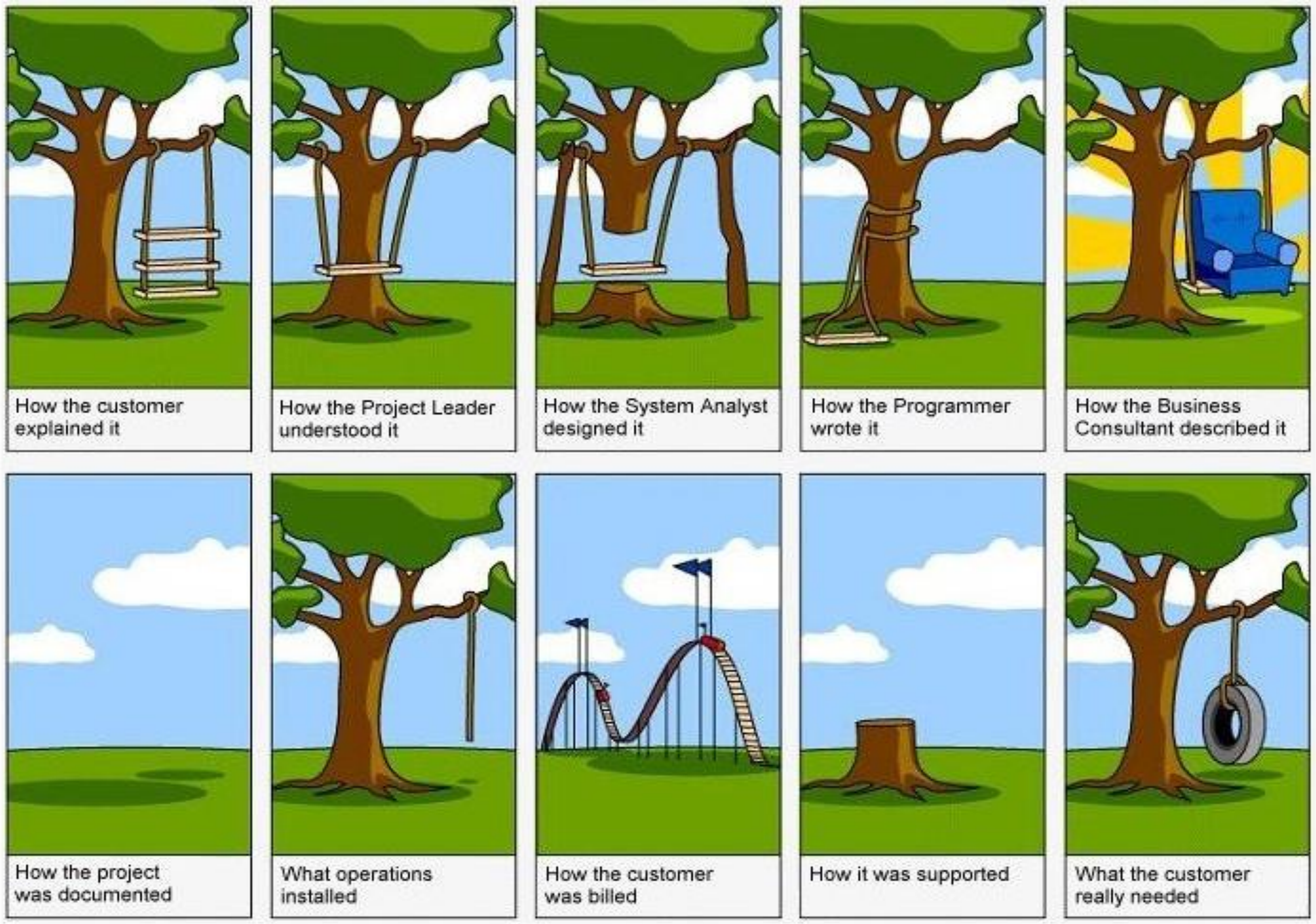
Process Discovery using Python (Praktikum)

Meeting 1 - Requirements Analysis, Scrum Overview

Madhavi Shankar, Mahsa Pourbafrani

Lab WS2020-2021

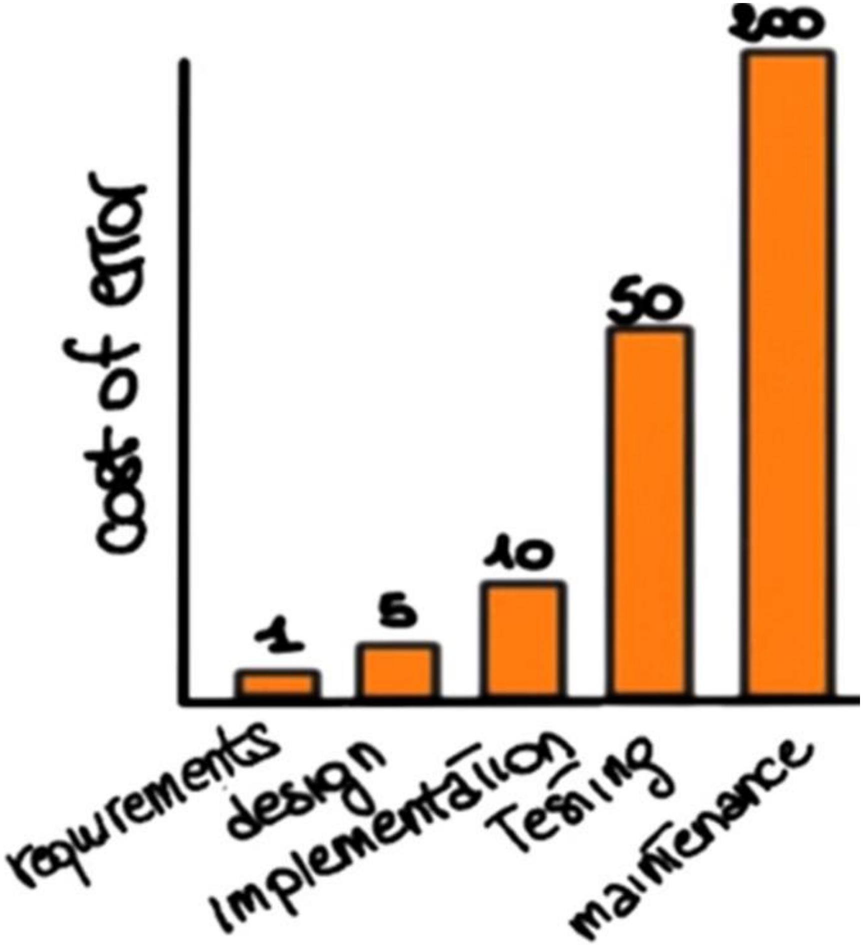
Why doing Requirements Analysis


How the customer explained it
How the Project Leader understood it
How the System Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Pic from: https://medium.com/omarelgabrys-blog/requirements-engineering-introduction-part-1-6d49001526d3

How to tackle this problem?

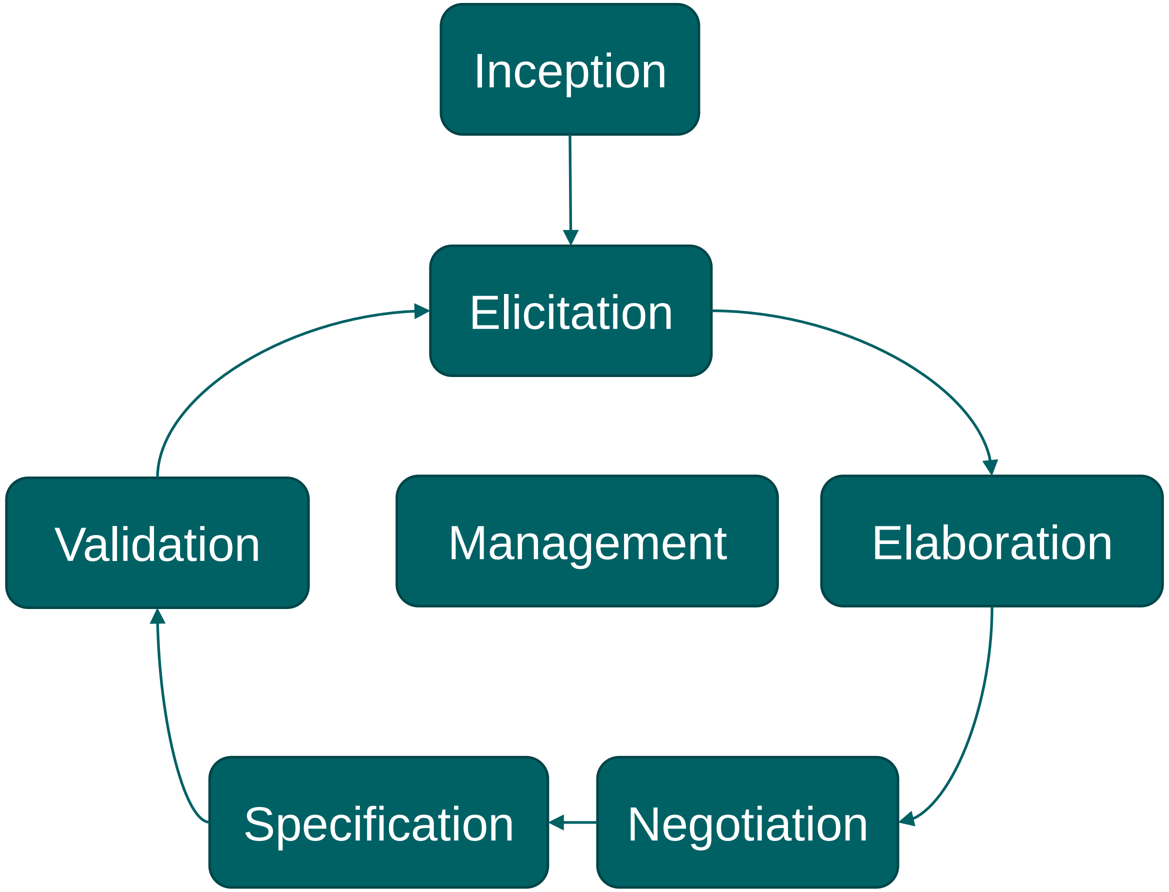
Why Requirements Analysis?



Error in requirement costs much higher in later stage

Stages in Requirements Engineering

- Inception
- Elicitation
- Elaboration
- Negotiation
- Specification
- Validation
- Management



Inception

- Ask a set of questions that establish …
  - basic understanding of the problem
  - Audience or users of the solution
  - the nature of the solution that is desired
  - the effectiveness of preliminary communication and collaboration between the customer and the developer

Start collecting information, and know your customer and other stackholders

Elicitation

- Elicit requirements from all stakeholders
  - Address problems of scope
  - Address problems of understanding
    - Customers are not sure about what is needed, skip “obvious” issues,
    - Have difficulty communicating with the software engineer,
    - Have a poor grasp of the problem domain
  - Address problems of volatility (changing requirements)
  - Sometimes we do discuss obvious technical stack selections.

Start answering problems related to scope, issues and make technical choices

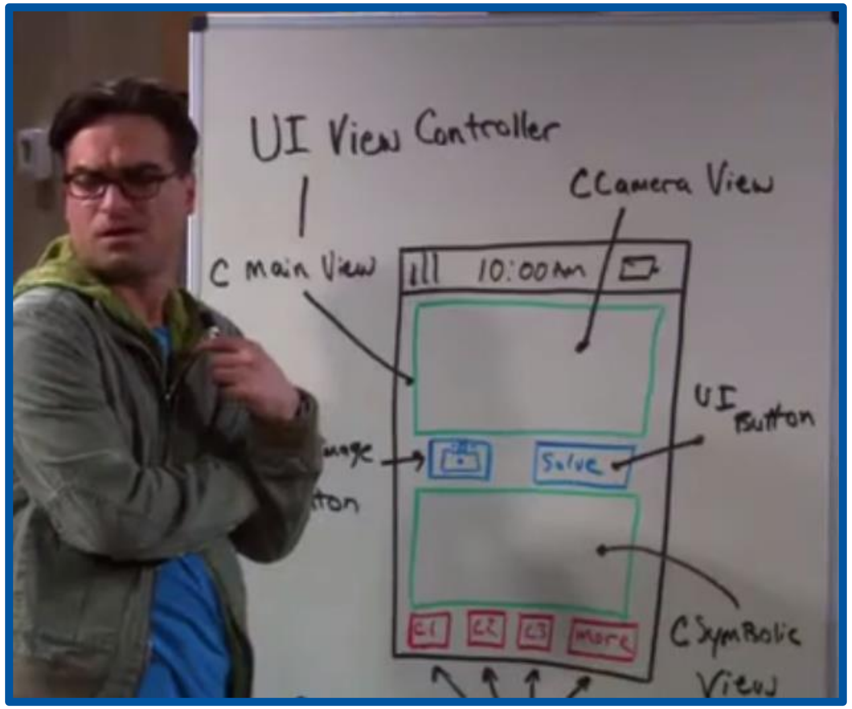
Elaboration and negotiation

- Elaboration: create an analysis model that identifies data, function, features, constraints and behavioral requirements
- Negotiation: agree on a deliverable system that is realistic for developers and customers
  - Rank requirements by priority (conflicts arise here …)
  - Identify and analyze risks associated with each requirement
  - “guestimate” efforts needed to implement each requirement
  - Eliminate, combine and / or modify requirements to make project realistic

Start designing and confirm

Specification

- Can be any one (or more) of the following:
  - A written document
  - A set of models
  - A formal mathematical model
  - A collection of user scenarios (use-cases)
  - A prototype



The Big Bang Theory Season 4 Episode 12

Design in more detail and concrete

Validation

- Review mechanism that looks for:
  - Errors in content or interpretation
  - Areas where clarification may be required
  - Missing information
  - Inconsistencies (a major problem when large products or systems are engineered)
  - Conflicting or unrealistic (unachievable) requirements
  - Tests for requirements

Confirm with your customer!!!

After confirmed, Email all stakeholders for leaving records

Management

- Involves managing change:
  - **Feature traceability:** how requirements relate to observable system/product features
  - **Source traceability:** identifies source of each requirement
  - **Dependency traceability:** how requirements are related to each other
  - **Subsystem traceability:** categorizes requirements by the sub system (s) they govern
  - **Interface traceability:** how requirements relate to both external and internal system interfaces

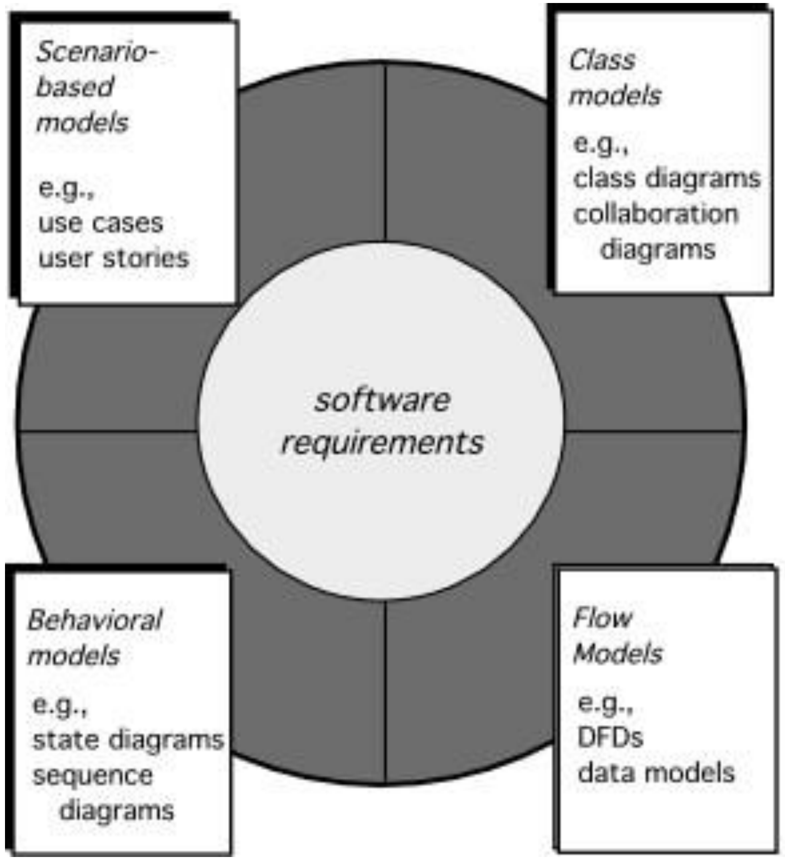
Keep things under control and document every change !!!

Methods and Tools

Many of them available

- Lists
  - elicitation question list
  - checklists for validation
- Graphical diagrams, good for communication
- Formal methods
  - e.g. UML for elaboration and specification

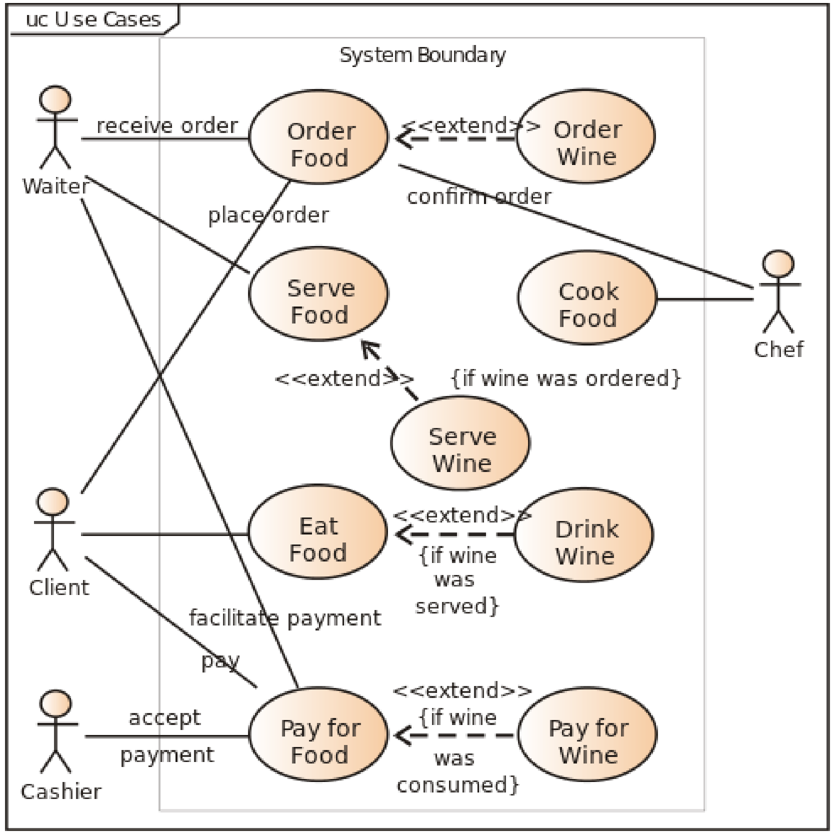
Modeling approaches



- Scenario-based: user interaction with system
  - Use-case: descriptions of the interaction
- Data-based: data objects in system
  - ER (entity-relation) Diagrams
  - Class-based: data + methods
    - OO, Class diagrams, implied by scenarios
- Behavioral-based: how external events change system
  - State, sequence diagram
- Flow-based: information transforms
  - Data flow, control flow diagrams

Use-case Diagram

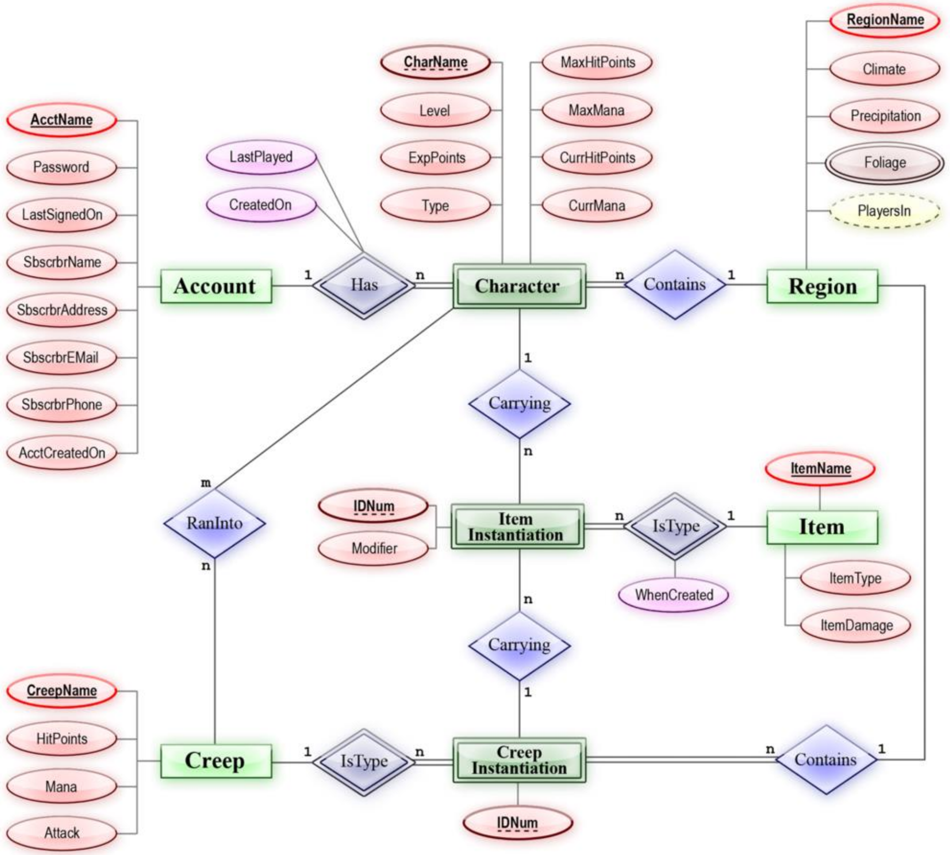
- Which role do which business(could access which function)



https://en.wikipedia.org/wiki/File:Use_case_restaurant_model.svg

ER (entity-relation) Diagrams

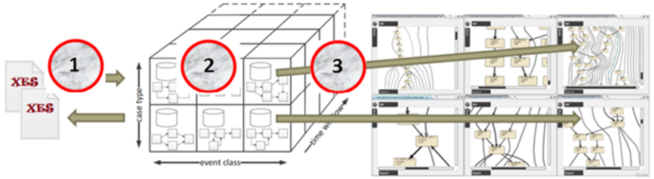
- Entity≈data object, do this before you design data structure for DB

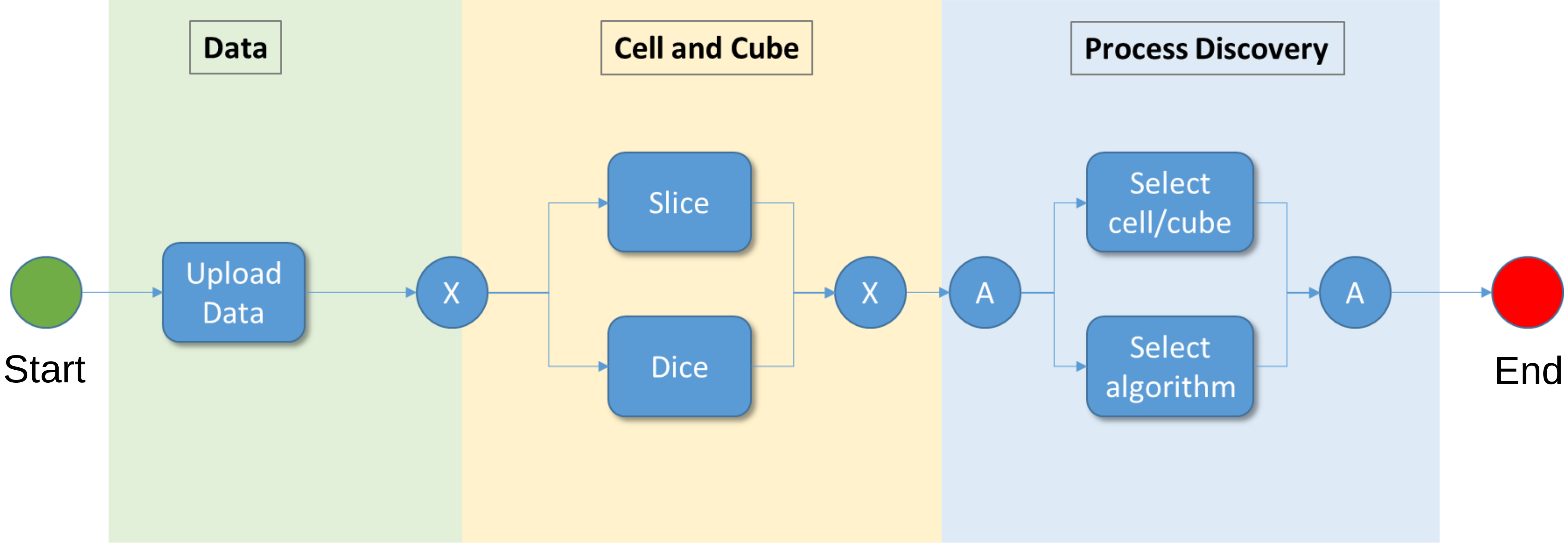


https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model

Function model

- BPMN, modeling the relation between functions, each function need to refer to certain requirement





Data

Cell and Cube

Process Discovery

Start

Upload Data

X

Slice

Dice

X

A

Select cell/cube

Select algorithm

A

End

Modeling for WebApps

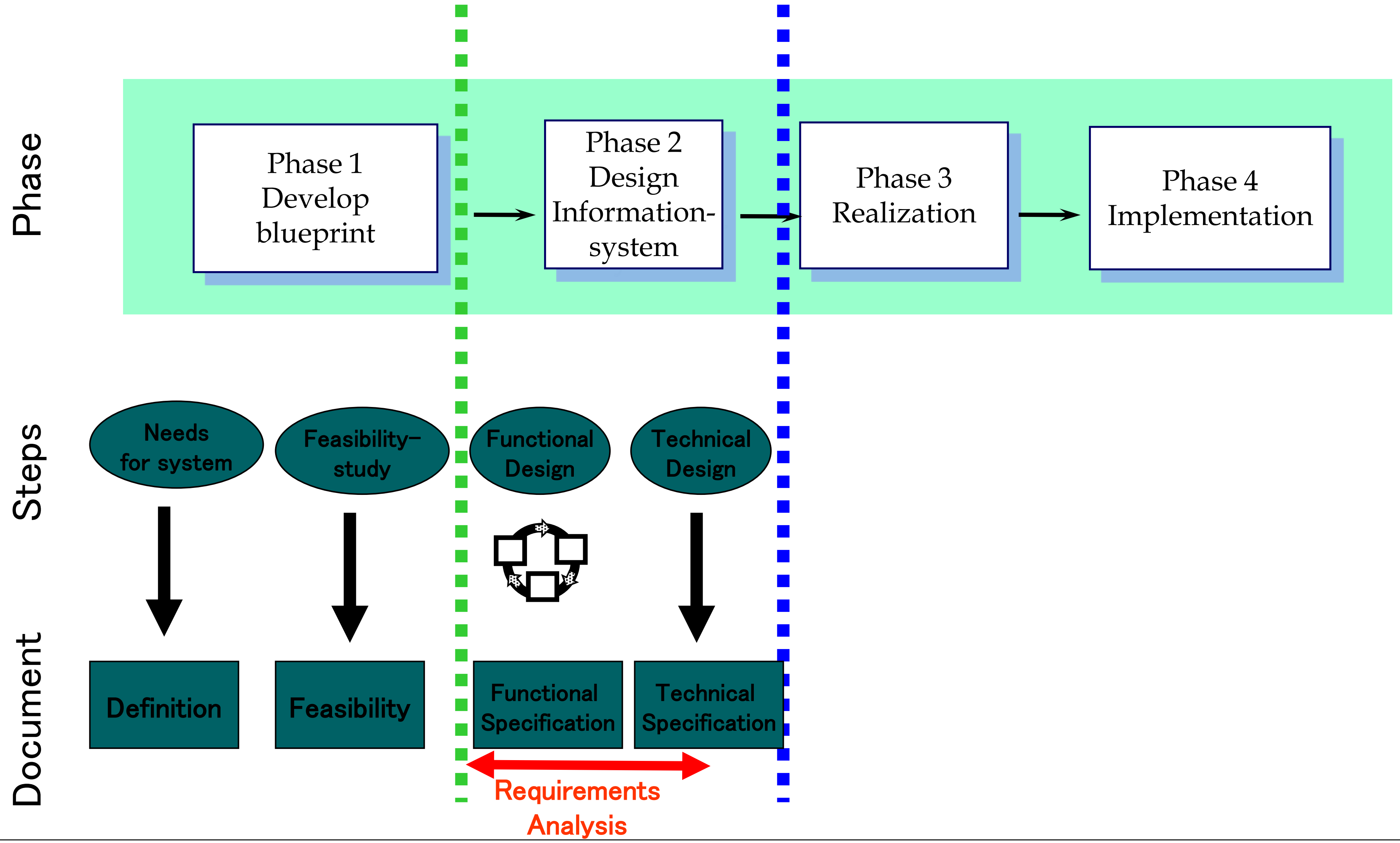
Content Analysis. the content is identified, including text, graphics and images, video, and audio data.

Interaction Analysis. The manner in which the user interacts with the WebApp is described in detail.

Functional Analysis. The usage scenarios (use-cases) created as part of interaction analysis define the operations that will be applied to WebApp content and imply other processing functions.

Configuration Analysis. The environment and infrastructure in which the WebApp resides are described in detail. Server-client side.

In which phase of the whole process?



Attention points

- Focus on the external performance of the system towards the users
- Limitations in the environment should be well described

  (technical, number of users and usage, etc.)
- Ease of adaptation
- References for maintaining the system
- Vision on the lifecycle of the system
- How to deal with unexpected events (fault-proof)

Templates for Requirement Specification Document

The IEEE standard for the Requirements Specification document is

https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc

Feel free to remove the unwanted sections. The sections mentioned in the document are for a proper development project.

End of Requirement Analysis

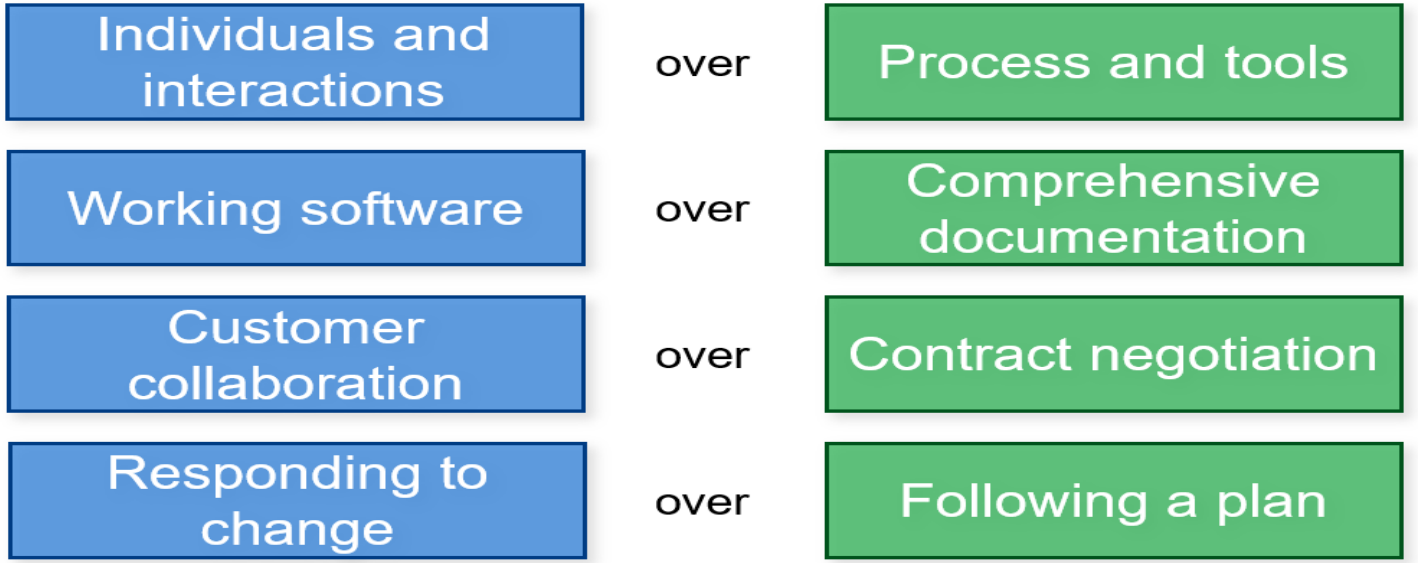
Q & A?

Agile Methodology

- Overview
- Introduction to Scrum

What is Agile Development?

Agile is a term used to describe software development approaches that employ continual planning, learning, improvement, team collaboration, evolutionary development, and early delivery.



Individuals and interactions over Process and tools

Working software over Comprehensive documentation

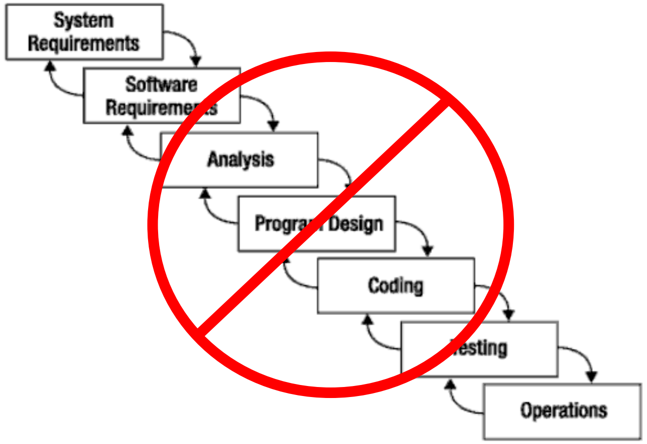
Customer collaboration over Contract negotiation

Responding to change over Following a plan

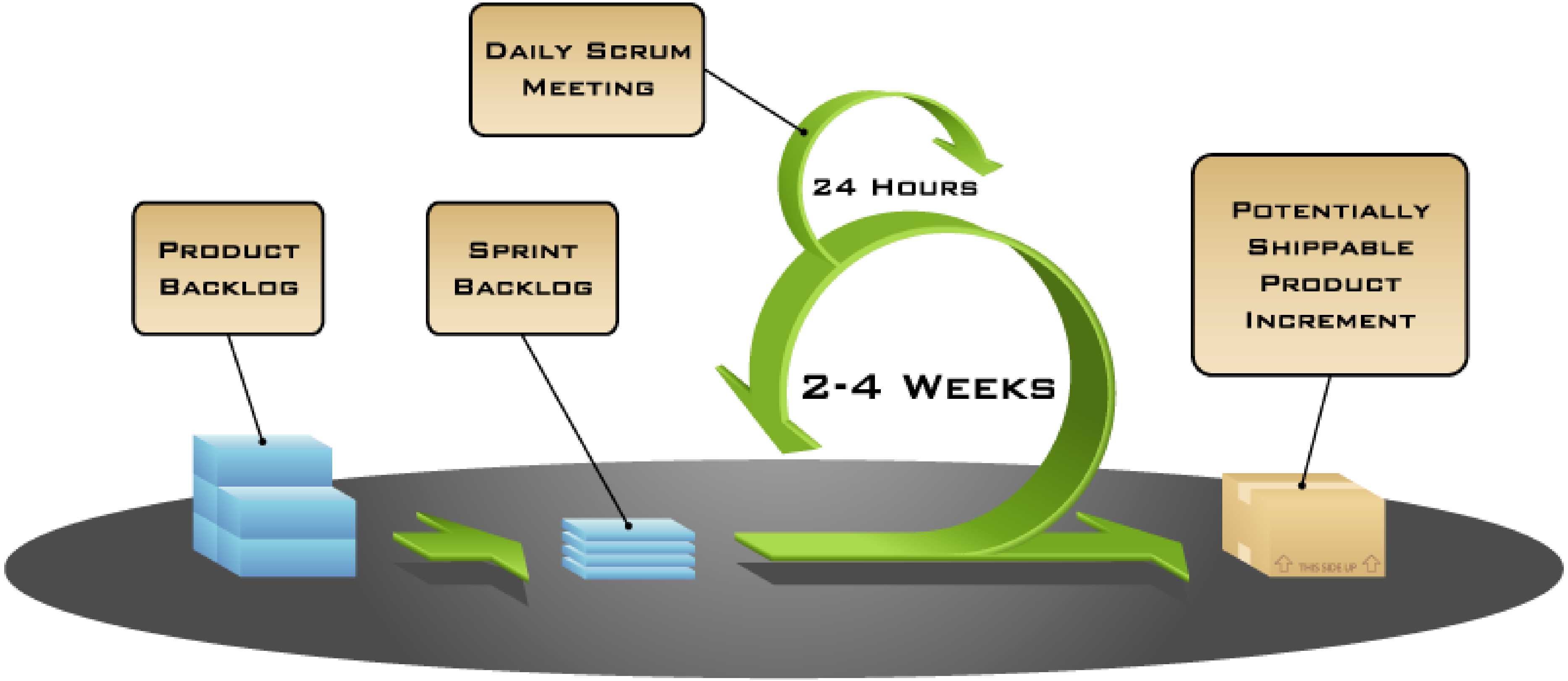
Source: www.agilemanifesto.org

What is Scrum?

- **Scrum**:
  - Is an agile, lightweight process
  - Can manage and control software and product development
  - Uses iterative, incremental practices
  - Increases productivity as less documentation and process overhead!
  - Reduces time to benefits
  - Embraces adaptive, empirical systems development
  - Is not restricted to software development projects

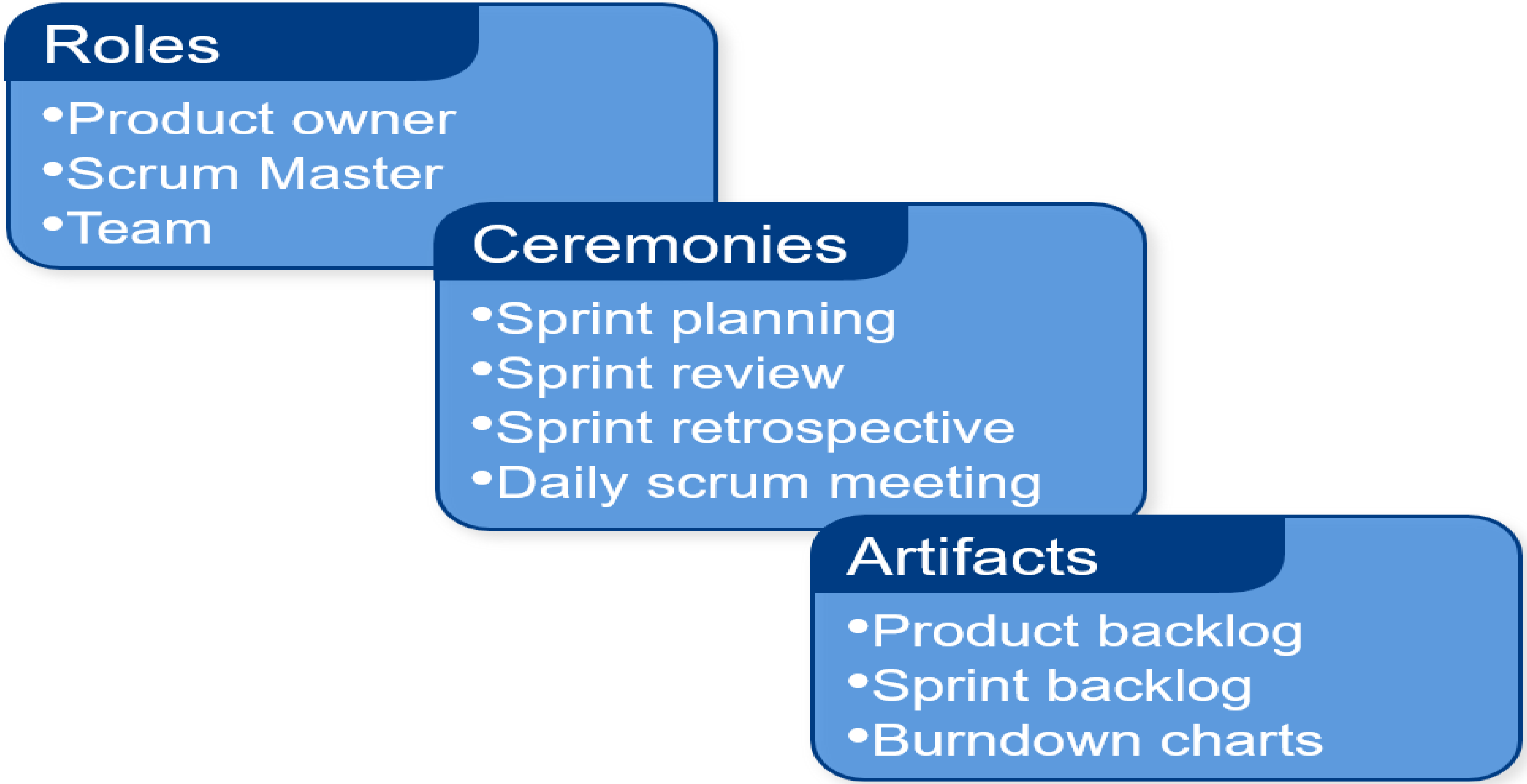


How SCRUM works?



Copyright © 2005, Mountain Goat Software

Scrum Framework



Roles

- Product owner
- Scrum Master
- Team

Ceremonies

- Sprint planning
- Sprint review
- Sprint retrospective
- Daily scrum meeting

Artifacts

- Product backlog
- Sprint backlog
- Burndown charts

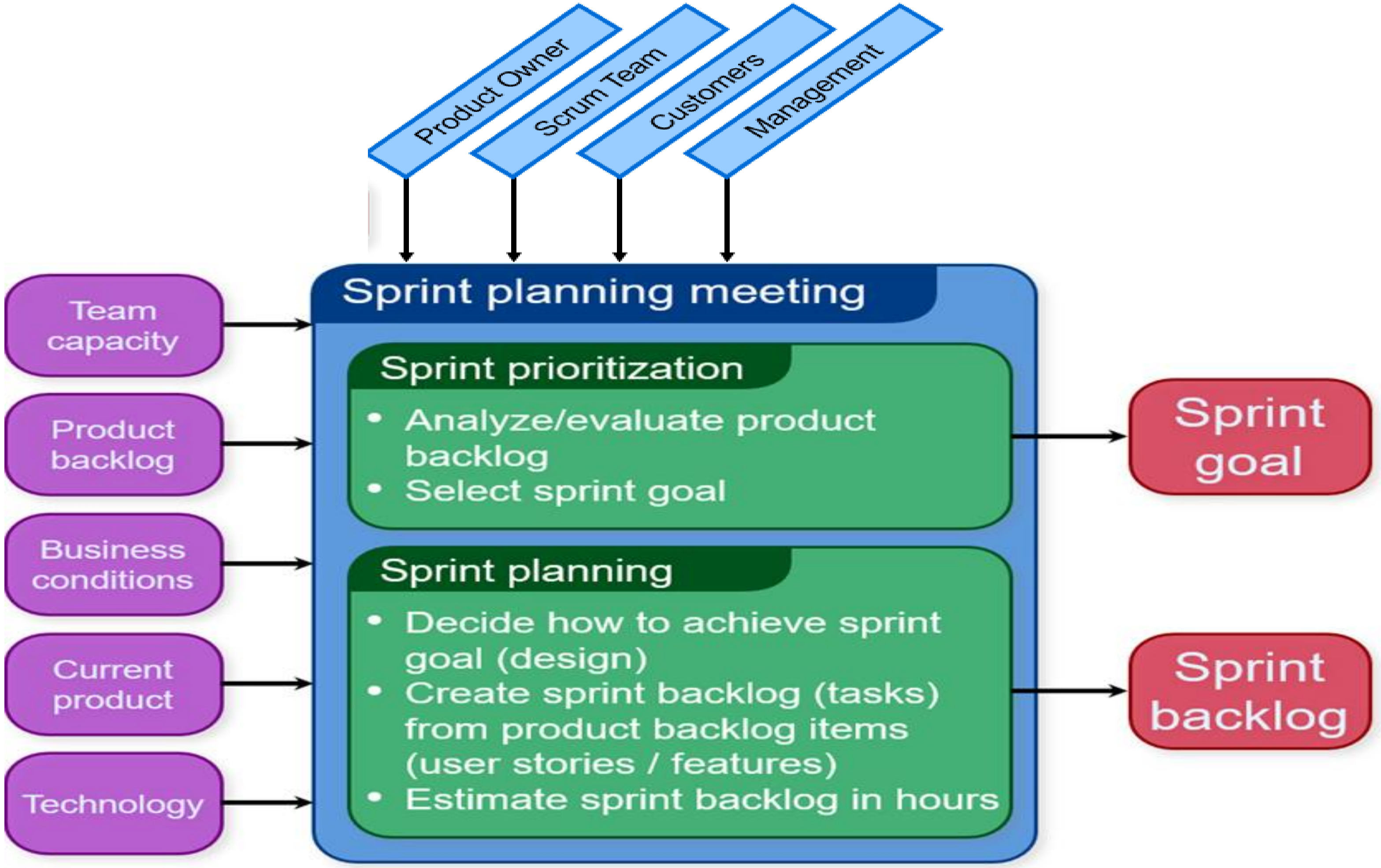
Scrum Roles

- Product Owner
  - Possibly a Product Manager or Project Sponsor
  - Decides features, release date, prioritization, $$$

- Scrum Master
  - Typically a Project Manager or Team Leader
  - Responsible for enacting Scrum values and practices
  - Remove impediments/politics, keeps everyone productive

- Project Team
  - 5-10 members; Teams are self-organizing
  - Cross-functional: QA, Programmers, UI Designers, etc.
  - Membership should change only between sprints

Sprint Planning Meeting



Product Owner

Scrum Team

Customers

Management

Team capacity

Product backlog

Business conditions

Current product

Technology

Sprint planning meeting

Sprint prioritization

- Analyze/evaluate product backlog
- Select sprint goal

Sprint planning

- Decide how to achieve sprint goal (design)
- Create sprint backlog (tasks) from product backlog items (user stories / features)
- Estimate sprint backlog in hours

Sprint goal

Sprint backlog

Daily Scrum Meetings

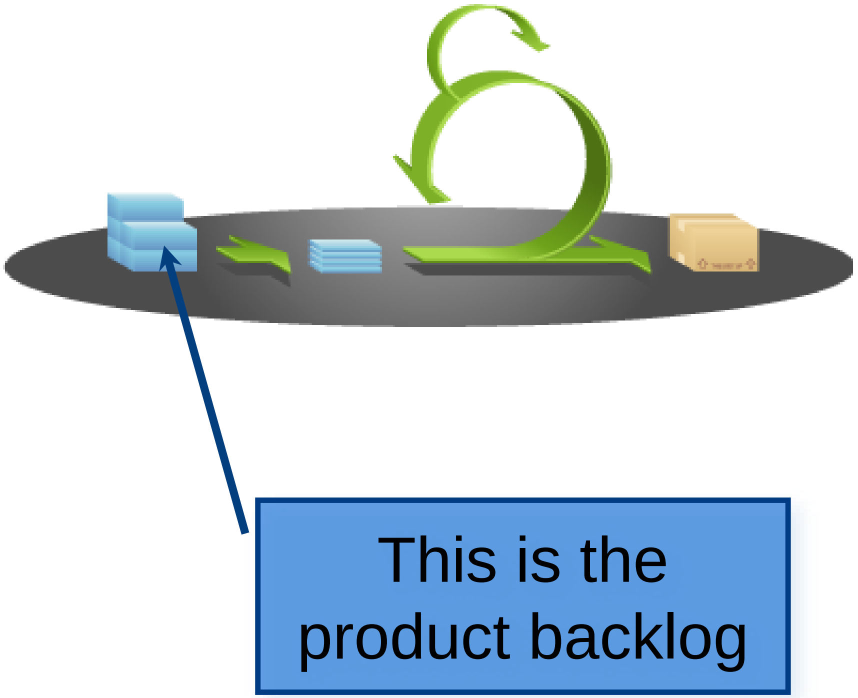
- Parameters
  - Daily, ~15 minutes, Stand-up
  - Anyone late pays a $1 fee – some discipline rules!
- Not for problem-solving
  - The whole world is invited
  - Only team members, Scrum Master, product owner, can talk
  - Helps avoid other unnecessary meetings
- Three questions answered by each team member:
  1. What did you do yesterday?
  2. What will you do today?
  3. What obstacles are in your way?

Scrum's Artefact

- Scrum has remarkably few artifacts
  - Product Backlog
  - Sprint Backlog
  - Burndown Charts

- Can be managed using just an Excel spreadsheet
  - More advanced/complicated tools exist:
    - Expensive
    - Web-based – no good for Scrum Master/project manager who travels
    - Still under development

Project Backlog

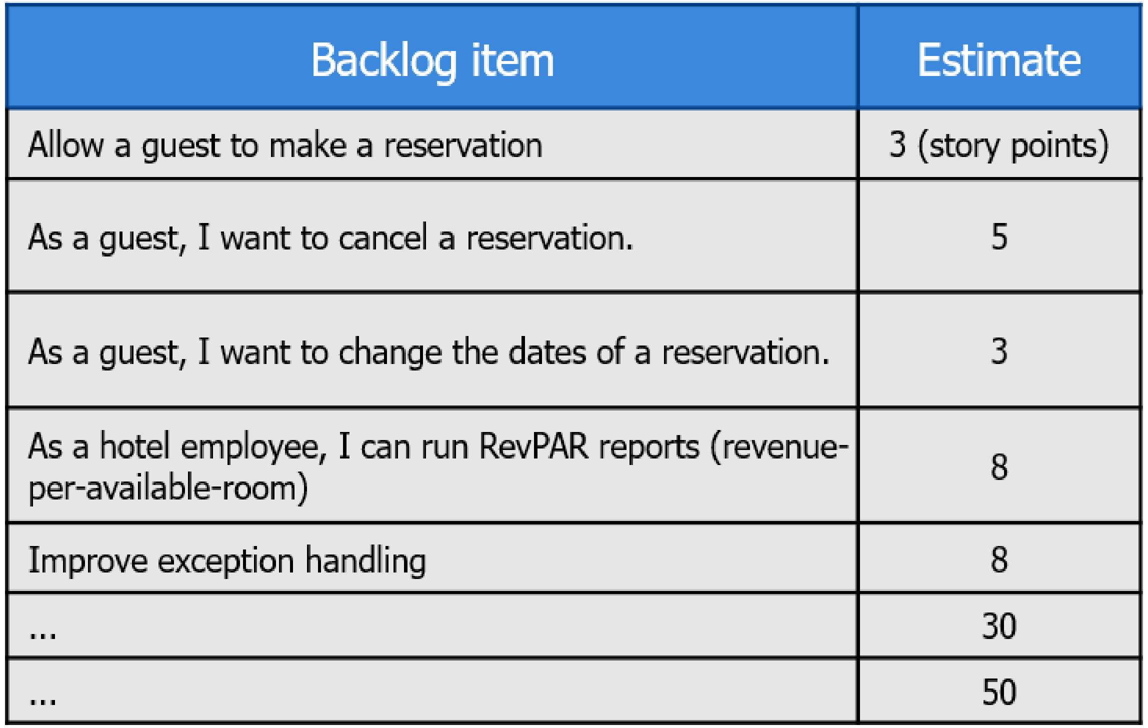


This is the product backlog

- ➢ The requirements, a list of all desired work on the project
- ➢ Prioritized by the product owner
- ➢ Reprioritized at the start of each sprint
- ➢ Can be any one of the 4 items
  - ➢ User Stories
  - ➢ Technical Stories – not exactly the main part, but also important like “update test tools”, “update software”
  - ➢ Spikes – POC, research, prototypes (time-bound)
  - ➢ Defects

User Stories

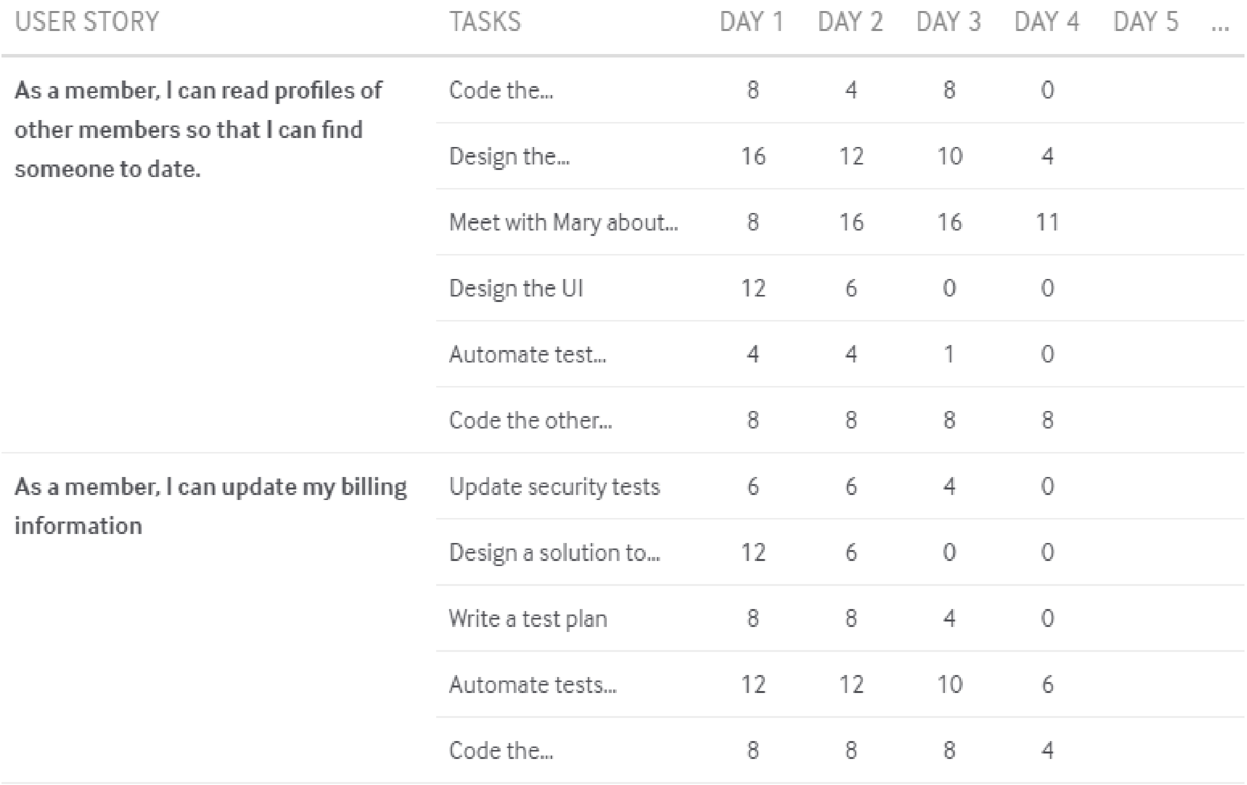
- Instead of Use Cases, Agile project owners do "user stories"
  - **Who** (user role) – Is this a customer, employee, admin, etc.?
  - **What** (goal) – What functionality must be achieved/developed?
  - **Why** (reason) – Why does the user want to accomplish this goal?
    - As a [user role], I want to [goal], so I can [reason].
- Example:
  - "As a hotelier, I want an online reservation portal, so the guests can make reservations."
- **story points**: Rating of effort needed to implement this story
  - common scales: 1-10, shirt sizes (XS, S, M, L, XL), etc.



Backlog item	Estimate
Allow a guest to make a reservation	3 (story points)
As a guest, I want to cancel a reservation.	5
As a guest, I want to change the dates of a reservation.	3
As a hotel employee, I can run RevPAR reports (revenue-per-available-room)	8
Improve exception handling	8
...	30
...	50

Sprint Backlog

- Individuals sign up for work of their own choosing
  - Work is never assigned
- Any team member can add, delete change sprint backlog
- Work for the sprint emerges
- If work is unclear, define a sprint backlog item with a larger amount of time and break it down later
- Update work remaining as more becomes known
- Estimated work remaining is updated daily



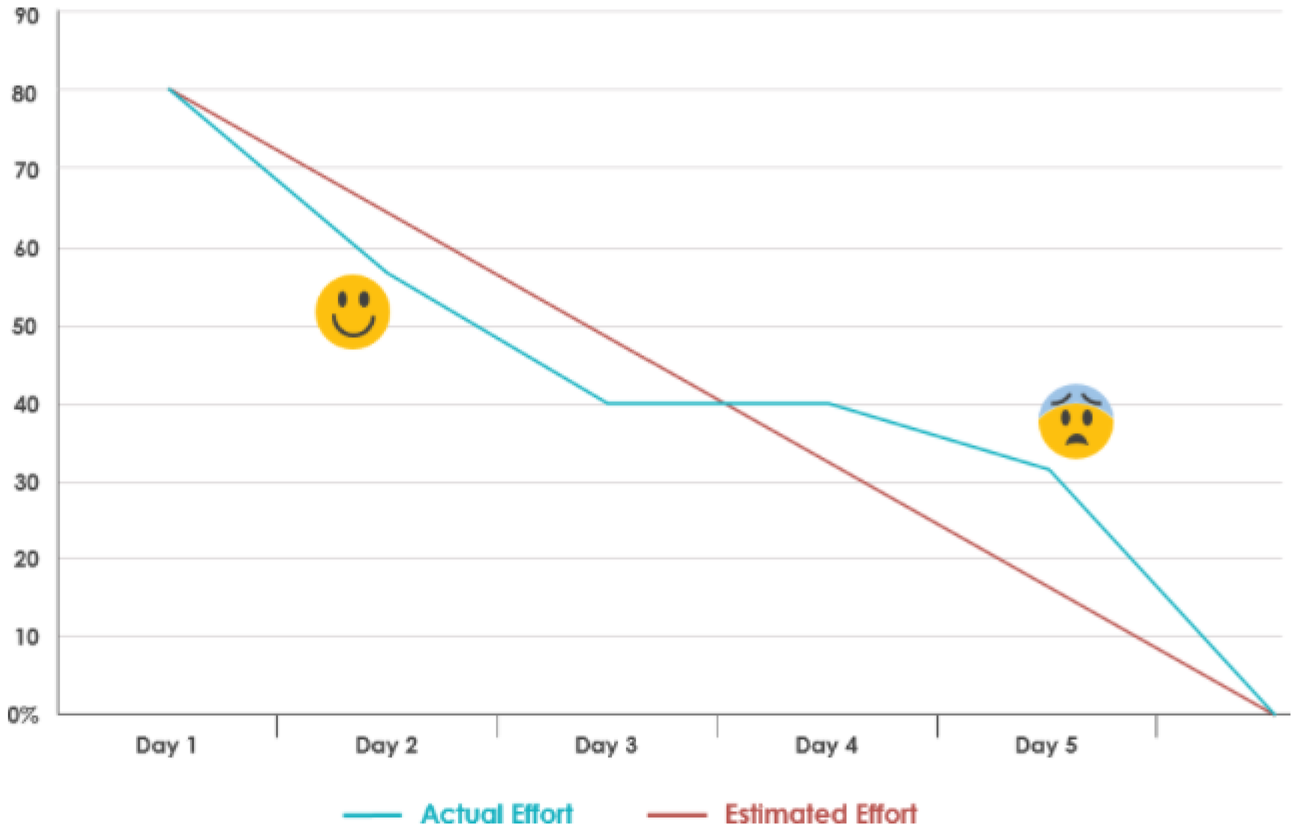
USER STORY	TASKS	DAY 1	DAY 2	DAY 3	DAY 4	DAY 5	...
As a member, I can read profiles of other members so that I can find someone to date.	Code the...	8	4	8	0		
	Design the...	16	12	10	4		
	Meet with Mary about...	8	16	16	11		
	Design the UI	12	6	0	0		
	Automate test...	4	4	1	0		
	Code the other...	8	8	8	8		
As a member, I can update my billing information	Update security tests	6	6	4	0		
	Design a solution to...	12	6	0	0		
	Write a test plan	8	8	4	0		
	Automate tests...	12	12	10	6		
	Code the...	8	8	8	4		

Sprint Burndown charts

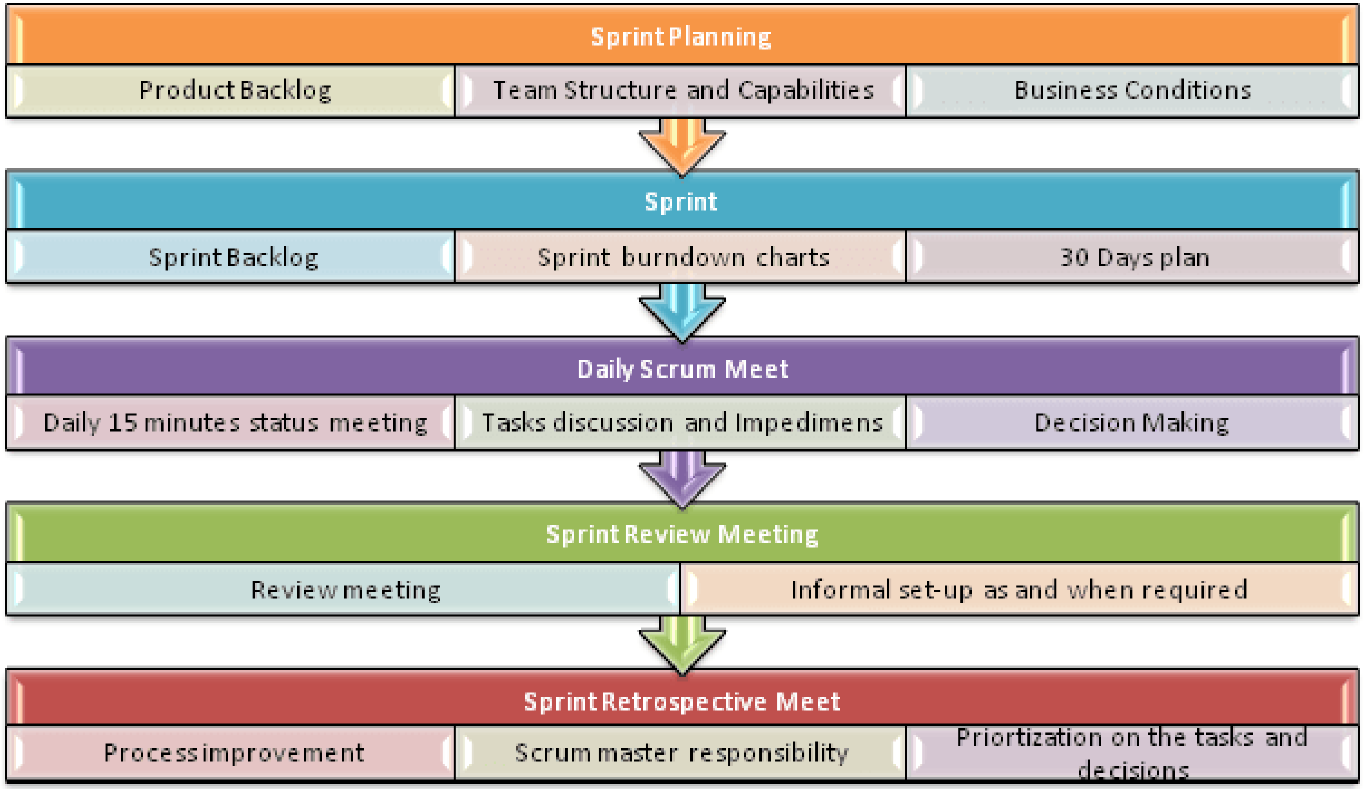
A burn-down chart is a graphical representation of work left to do versus time.

This helps in:

- monitoring the project scope creep
- keeping the team running on schedule
- comparing the planned work against the team progression



Sprint overview summary



Sprint Planning

Product Backlog | Team Structure and Capabilities | Business Conditions

Sprint

Sprint Backlog | Sprint burndown charts | 30 Days plan

Daily Scrum Meet

Daily 15 minutes status meeting | Tasks discussion and Impedimens | Decision Making

Sprint Review Meeting

Review meeting | Informal set-up as and when required

Sprint Retrospective Meet

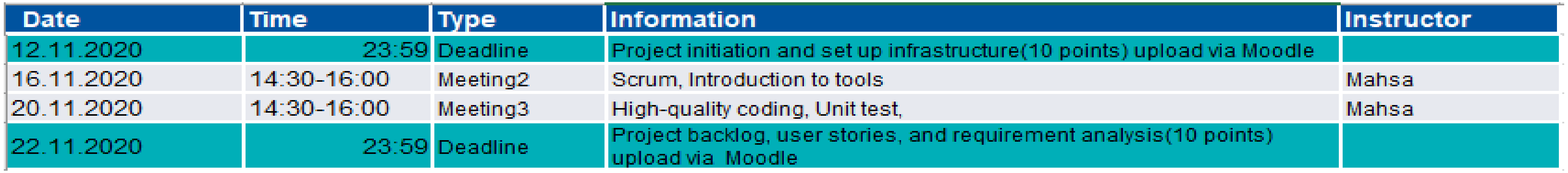
Process improvement | Scrum master responsibility | Priortization on the tasks and decisions

More details related to the following Scrum framework will be discussed further in the next meeting.

Expectations from the Phase

- Prepare a Requirement analysis document for your assignment.
- The document must contain:
  - Requirements Analysis
  - Mention which requirements tool would suit your approach best (Need not use any tool)
  - Decompose the high-level requirements to smaller modules and create smaller use cases
  - Use case Models. Use any UML representations.
- In Trello, create the User stories, associated Product Backlogs, and Sprint Backlogs.

Important Dates and Milestones at One Glance



Date	Time	Type	Information	Instructor
12.11.2020	23:59	Deadline	Project initiation and set up infrastructure(10 points) upload via Moodle	
16.11.2020	14:30-16:00	Meeting2	Scrum, Introduction to tools	Mahsa
20.11.2020	14:30-16:00	Meeting3	High-quality coding, Unit test,	Mahsa
22.11.2020	23:59	Deadline	Project backlog, user stories, and requirement analysis(10 points) upload via Moodle	

Thank You!