



# SOFTWARE ENGINEERING

## Lecture 2: Software Engineering Processes / Software Development Lifecycle



Prof. Mohamad Kassab    [m.kassab@nyu.edu](mailto:m.kassab@nyu.edu)

# OUTLINE

**I. How do we start a software project?**

**II. Stakeholders Identifications**

**III. Software Process Models**

- The Waterfall Model
- Prototyping Model
- Agile Methods



A group of people are gathered around a table in a meeting. One person is pointing at a document on the table, which contains various charts and graphs. A laptop is open on the table, and a green mug is visible. The scene is brightly lit, suggesting an indoor office environment.

## I. HOW DO WE START A SOFTWARE PROJECT?



# INITIATING A SOFTWARE PROJECT

- Starting a software project involves defining the “Business Goal” as a first step.

- **Star Trek (original):**

To explore strange new worlds, to seek out new life forms and new civilizations, to boldly go where no man has gone before.

- **Baggage Handling System:**

To automate all aspects of baggage handling from passenger origin to destination.

# BUSINESS GOALS

- Organizations have a reason for needing the proposed system
- This could be characterized in a number of different ways
  - Market intent
  - Product roadmap
  - Business & operational strategy
  - ...
- Ideally this would describe high level goals, associated measures, and tactical approaches



# TYPES OF ORGANIZATIONS

- There are several types of organizations
- We'll look at a couple of examples here:
  1. Those that generate revenue directly from the construction, sale or licensing of software intensive products.
  2. Those that generate revenue from non-software sales and services but rely on software for operations.



# SOFTWARE ORIENTED COMPANIES

- If the organizations is in the business of developing software they will often have some kind of product roadmap
- This roadmap typically includes things like:
  - Market related information
  - Value proposition for the market
  - Competitive analysis
  - Anticipated number of units
  - Target price
  - Anticipated revenue
  - Associated budgets
  - ...



# NON-SOFTWARE COMPANIES

- These organizations often have strategic plans.
- These plans are focused on their domain and not specifically on software.
- You can be sure, however, that the objectives reflected in the plan rely on software.
  - Even if no one realizes that's the case
- Consider **Walmart**





# WALMART AND DATA



- Walmart collects data on every transaction that every customer makes in every store
- This is roughly 267 million transactions a day
- They are forever trying to improve their understanding of buying characteristics
- This knowledge can influence
  - Promotions
  - Inventory
  - Supply chain process
  - Store layout
  - ...
- Think about what goals in this area would mean to the IT systems ...



# CASE STUDY: BUILDING AUTOMATION SYSTEM



- Consider a company that primarily sells hardware devices for building automation.
- They have software applications that manage a network of these devices but this constitutes a loss leader, i.e. they lose money on the software but it helps the sale of the hardware devices.
- The company realizes that the hardware is being commoditized and over time the profit margins on the sale of their hardware devices are going to shrink. In order to sustain their business long term, the company decides to create a new building automation system that will be profitable.





# CASE STUDY: BUILDING AUTOMATION SYSTEM



- They wish to accomplish this by doing two things

**reduce internal development costs** and **expand the market.**

- **Internal development costs** can be reduced by replacing several of the existing applications with a single building automation system.
- **Market expansion** can be achieved by entering new and emerging geographic markets and opening new sales channel in the form of Value Added Resellers (VARs). VARs sell the new building automation system under their own brand to support building automation hardware devices of many different manufacturers



# **CASE STUDY:** BUILDING AUTOMATION SYSTEM

- **Business goals for Building Automation System:**
  - **Reduce internal development costs**
  - **Expand by entering new and emerging geographic markets**
  - **Open new sales channels in the form of value added resellers (VARs)**





A network diagram illustrating stakeholder identification. It features seven wooden figures of various colors (red, yellow, orange, blue, green, light green, and pink) arranged on a light-colored surface. The figures are connected by a network of grey ropes, forming a web of relationships. The connections are as follows: the red figure is connected to the yellow figure; the yellow figure is connected to the orange figure; the orange figure is connected to the blue figure; the blue figure is connected to the green figure; the green figure is connected to the light green figure; the light green figure is connected to the pink figure; and the blue figure is also connected to the light green figure. The ropes are thick and grey, and the figures are made of wood with a smooth finish.

## II. STAKEHOLDERS IDENTIFICATION

# REQUIREMENTS STAKEHOLDERS

- Who are they?
- What do they want?
- What don't they want?
- Do they really know what they want?
- Why do they change their minds?
- How can you uncover and mitigate these issues?



# WHO IS A STAKEHOLDER?



- A customer (or client or user)?
- The customer is always right, but there are more persons/entities with an interest in the system.
- We use “stakeholder” to remember that others, not just the customer are involved.

# TYPICAL STAKEHOLDERS

Customers (clients,  
users)

Sponsors (those who  
have commissioned  
and/or will pay for the  
system)

All responsible engineering  
and technical persons (e.g.  
systems, development, test,  
maintenance)

Regulators (typically,  
government agencies  
at various levels)

Third parties that have an  
interest in the system but no  
direct regulatory authority  
(e.g. standards organizations,  
users groups)

Society (is the system  
safe?)

Environment (for  
physical systems)

Professors (e.g. for  
this course)

This is an incomplete  
list





# TYPICAL STAKEHOLDERS (CONTINUE)

Those who  
“lose” if  
system is built

Competitors

Activists

Enemies

Gadflies

# STAKEHOLDERS IDENTIFICATION QUESTIONNAIRES (TO ASSIST IN IDENTIFYING STAKEHOLDERS)

- Who is paying for the system?
- Who is going to use the system?
- Who is going to judge the fitness of the system for use?
- What agencies (government) and entities (non-government) regulate any aspect of the system?
- What laws govern the construction, deployment and operation of the system?
- Who is involved in any aspect of the specification, design, construction, testing, maintenance, and retirement of the system?
- Who will be negatively affected if the system is built?
- Who else cares if this system exists or doesn't exist?
- Who have we left out?

## Exemplar System

- Airline baggage handling
  - Embedded, real-time
  - Moves baggage from a central loading point and redirects to one of many conveyors based on bag id
- Denver International Airport tried a such a system
- System used PCs, thousands of remote-controlled carts, 21-mile-long track
- Carts move along the track, carrying luggage from check-in counters to sorting areas and then straight to the flights waiting at airport gates.
- Each piece of baggage has a special bar-coded tag
- After spending \$230 million over 10 years, project was cancelled\*

<https://www5.in.tum.de/~huckle/DIABaggage.pdf>

# Airline Baggage Handling System

---

- **Who is paying for the system?** – Airline, grants, passengers, tax dollars?
- **Who is going to use the system?** – Airline personnel, maintenance personnel, travelers (at the end). Who else?
- **Who is going to judge the fitness of the system for use?** – Airline, customers, unions, FAA, OSHA, the press, independent rating agencies. Who else?
- **What agencies (government) and entities (non-government) regulate any aspect of the system?** – FAA, OSHA, union contracts, State and local codes. What else?
- **What laws govern the construction, deployment and operation of the system?** – Various state and local building codes, federal regulations for baggage handling systems, OSHA laws? What else?
- **Who is involved in any aspect of the specification, design, construction, testing, maintenance, and retirement of the system?** – various engineers, technicians, baggage handlers union, etc. We need to know them all.
- **Who will be negatively affected if the system is built?** – passengers? Union personnel? Who else?
- **Who else cares if this system exists or doesn't exist?** – limousine drivers? Who else?
- **Who have we left out?**





# STAKEHOLDER PRIORITIZATION

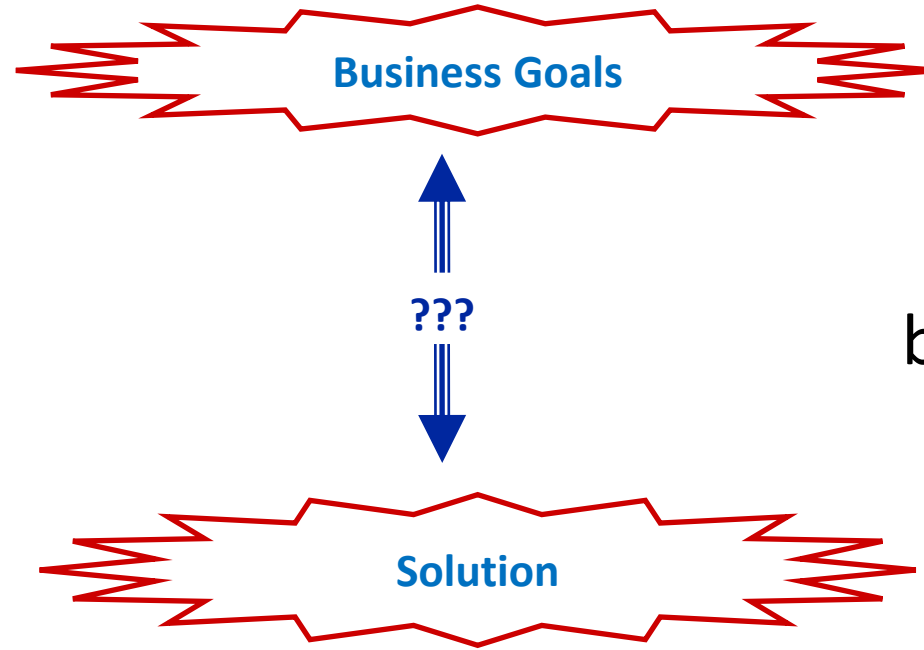
- Not all stakeholders are of equal importance.
- Stakeholder needs and desires may conflict.
- Ranking/prioritization helps in resolving these conflicts.
- Usually rank denotes risk of not satisfying the stakeholder (e.g. legal requirements should be #1 unless you want to go to jail)
- Requirements prioritization is the key to reconciliation and risk mitigation -- so get used to it!



# Partial Ranking for Baggage Handling System

<i>Stakeholder Class</i>	<i>Rank</i>	<i>Rationale</i>
System maintenance personnel	Medium	They have moderate interaction with the system.
Baggage handlers	Medium	They have regular interaction with the system but have an agenda that may run counter to the customer.
Airline schedulers/ dispatchers	Low	They have little interaction with the system.
Airport personnel	Low	Most other airport personnel have little interaction with the system.
Airport managers and policy makers ("the customer")	High	They are paying for the system.

# III. Software Process Models



How to bridge the gap  
between Business Goals and  
solutions?

24

---

# THE BIG PROBLEM







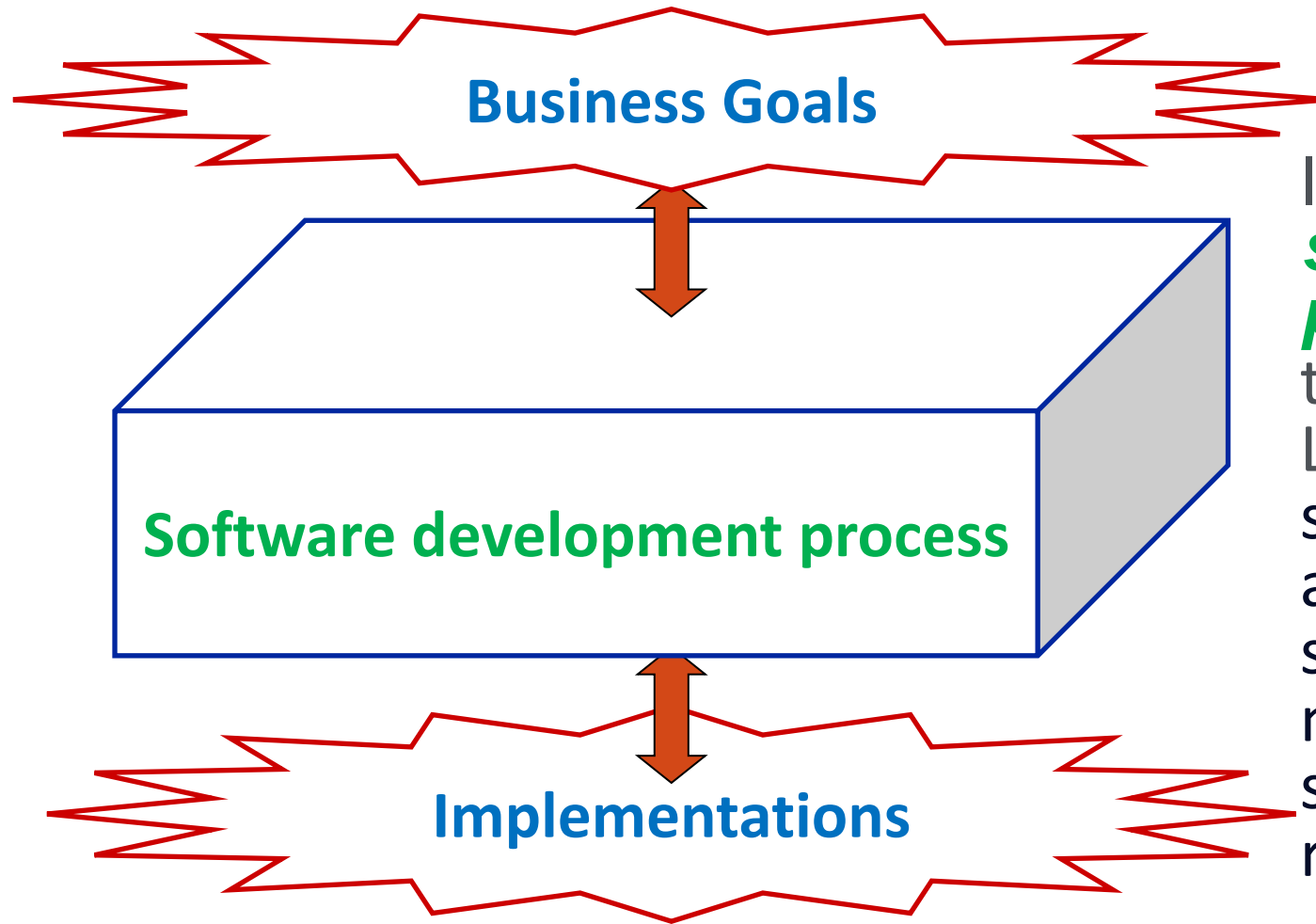
- Ad hoc
- Requires gurus
- Unpredictable
- Costly

25

---

# ONE POSSIBLE ANSWER





In software engineering, a **software development process**, also known as the Software Development Life Cycle (SDLC), includes several phases that provide a method for building software products that meet technical specifications and user requirements.

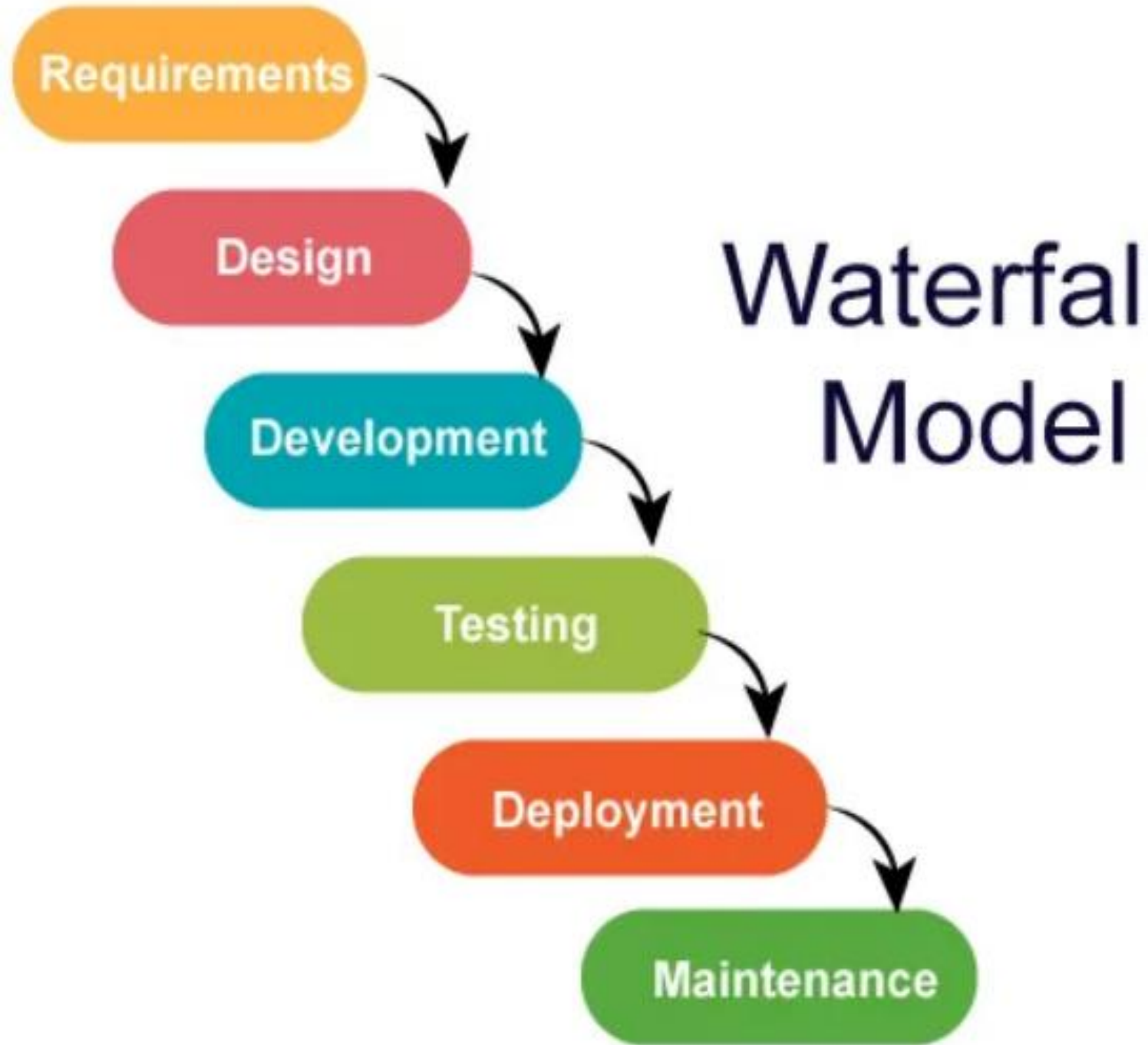
# PROCESS MODELS

- There are different process models. Let's explore some of these:
  - The Waterfall Model
  - Prototyping Model
  - Agile Method



# THE WATERFALL MODEL





- The Waterfall model is the oldest paradigm in software engineering.
- Still widely used, especially by organizations like the US Department of Defense and NASA.
- Emphasizes spending time early on to ensure the accuracy of requirements and design.
- Divides the project into distinct phases.
- Order of Phases: Requirements → Design → Implementation → Testing → Maintenance.

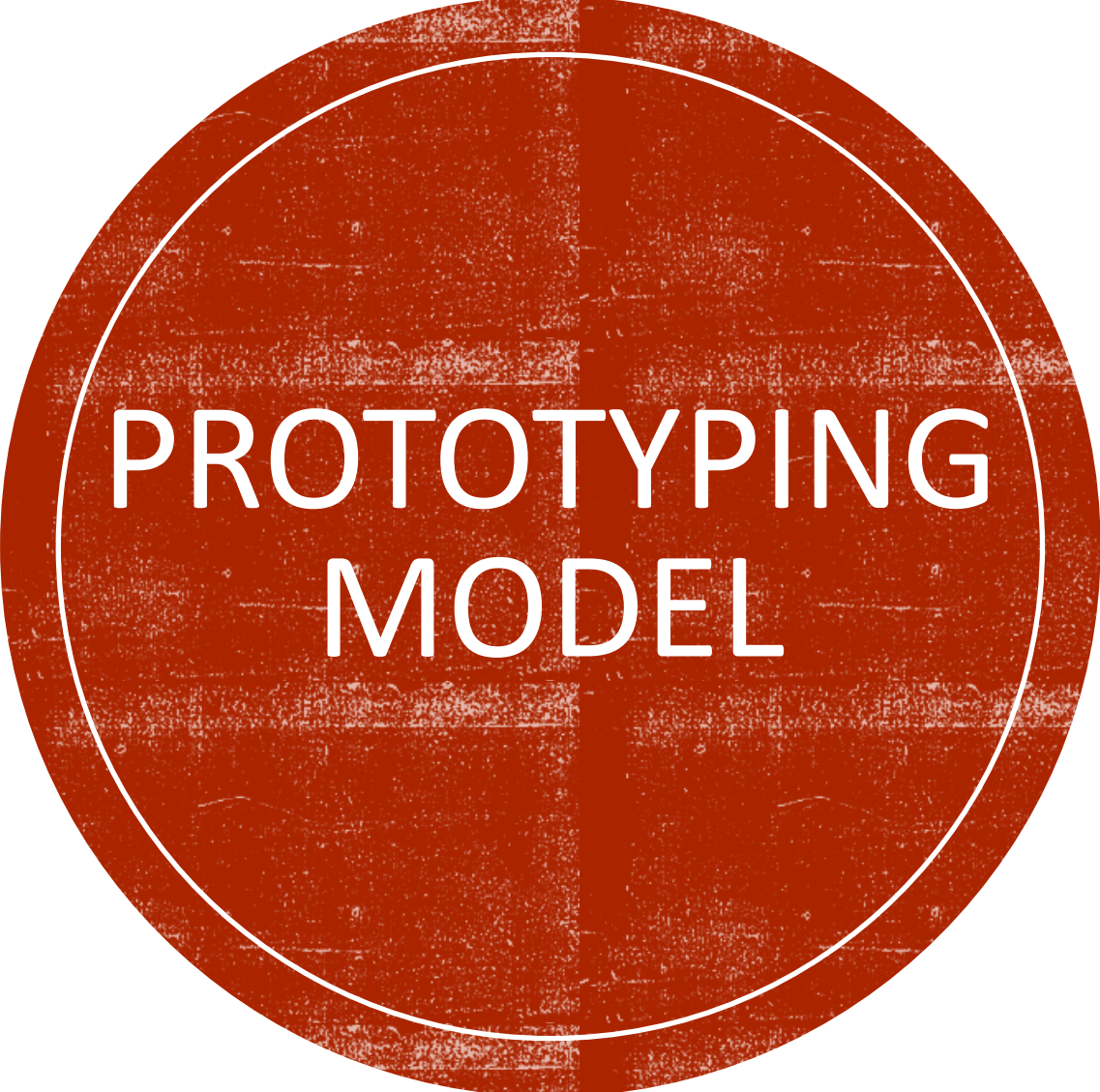


# APPLICABILITY OF WATERFALL MODEL


- Ideal for well-defined adaptations or enhancements to an existing system.
- Suitable for new development efforts when requirements are well defined and stable.
- Commonly used in government projects, large-scale systems with defined specifications.
- **NASA and DoD:** Noteworthy examples of organizations utilizing the Waterfall model.

# CHALLENGES WITH WATERFALL MODEL

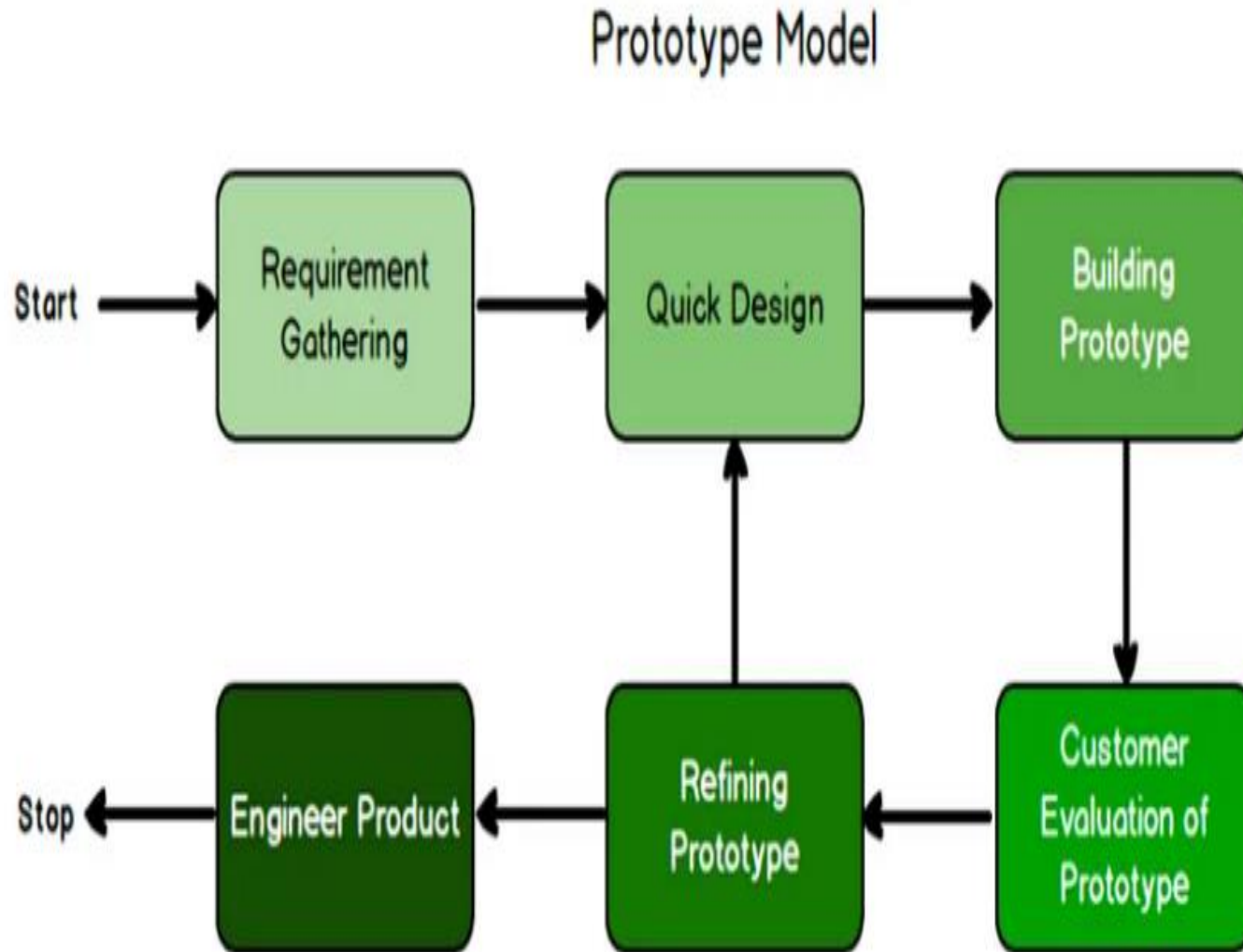
- **Inflexibility:** Does not accommodate changes well after the project has started.
- **Late Testing:** Testing occurs late in the process, raising the risk of identifying issues at a later stage.
- **Delayed Work Products:** Deliverables are not available until late in the project timeline.



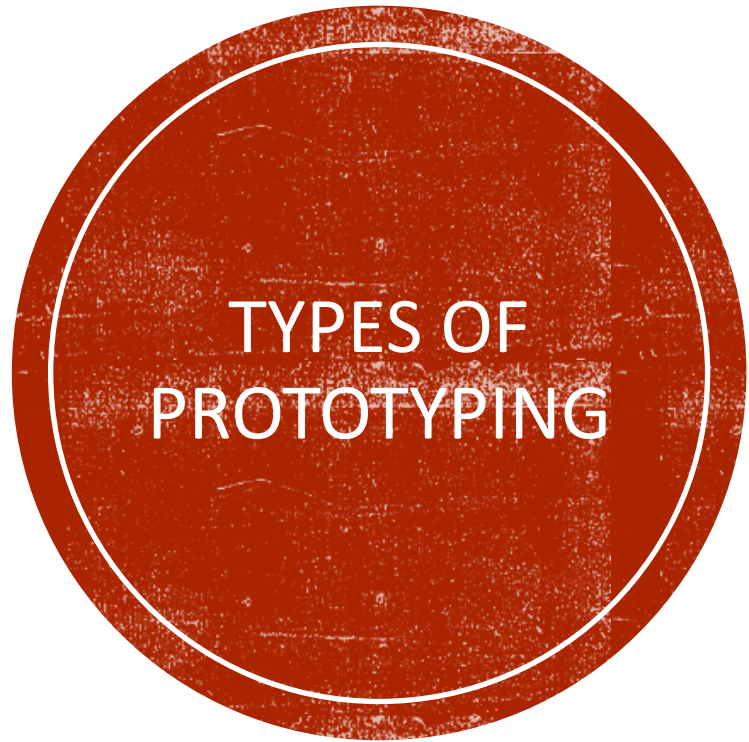
# PROTOTYPING MODEL



# PROTOTYPING MODEL

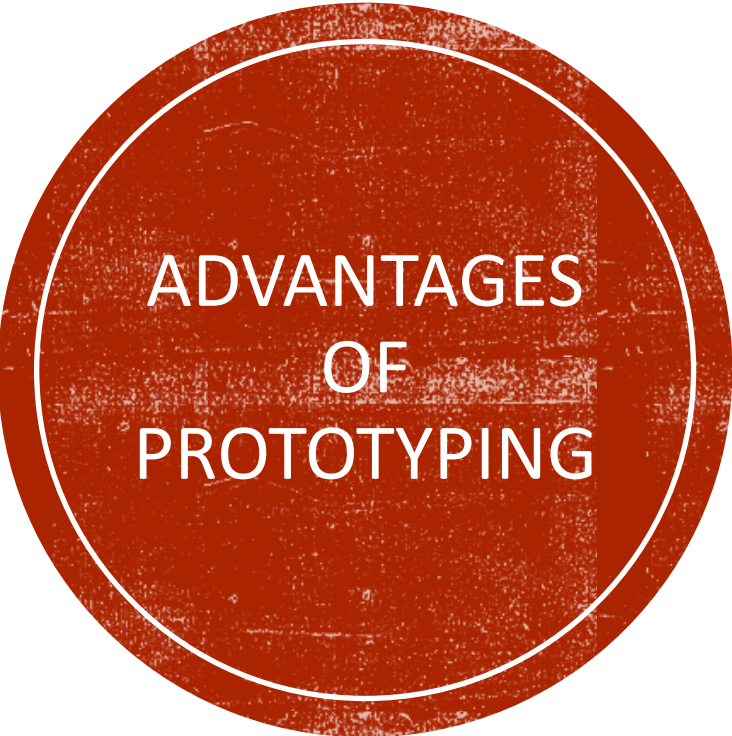


- *Prototyping* assists you and the customer to better understand what is to be built when requirements are fuzzy.
- When a customer just defines a set of general objectives for the software.
- A minimum viable product (**MVP**) can then be produced.



- **Throwaway/Rapid Prototyping:**  
Quickly built and discarded after obtaining feedback.
- **Evolutionary Prototyping:**  
Incremental development, refining the prototype based on user feedback.





## ADVANTAGES OF PROTOTYPING

- **User Involvement:** Encourages active participation and feedback from end-users.
- **Early Detection of Issues:** Identifies problems and changes early in the development process.
- **Flexible Changes:** Allows for changes in requirements during development.

# CHALLENGES OF PROTOTYPING

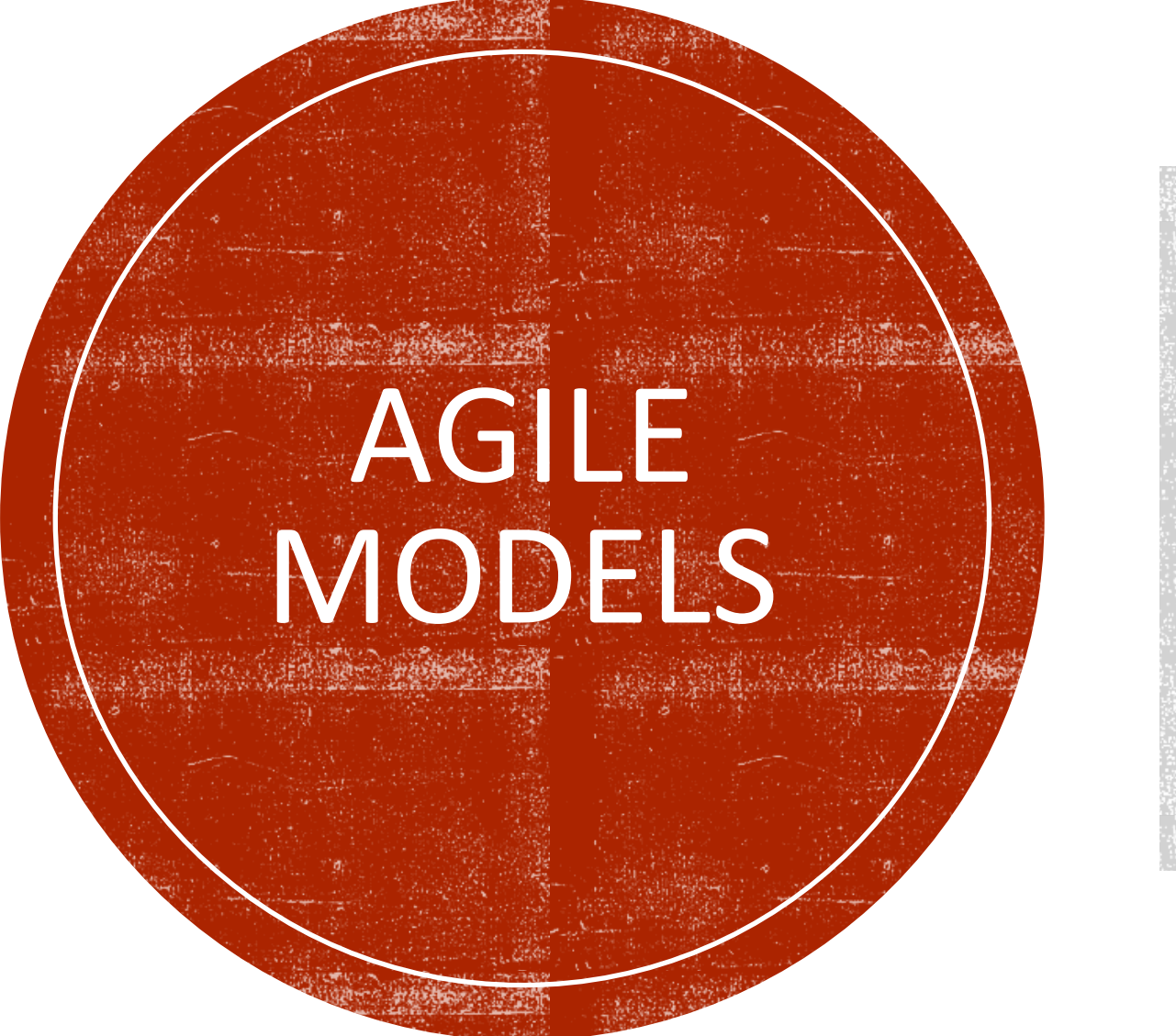
- **Time-Consuming:** Iterative nature may extend development time.
- **Incomplete Functionality:** Early prototypes might lack full functionality.
- **Miscommunication:** Potential for misunderstandings between developers and users.

# PROTOTYPING : REAL WORLD EXAMPLES AND TOOLS

- **Mobile App Development:** Prototyping is commonly used for mobile applications to refine user interfaces.
- **Web Development:** Rapid prototyping is effective for user experience testing in web development.
- **Balsamiq, Axure, Sketch:** Popular tools for creating wireframes and interactive prototypes.
- **Figma:** Collaboration platforms for sharing and testing prototypes.

# EXERCISE (OPTIONAL)

- **Objective:** Create a wireframe prototype for a weather application screen using Balsamiq.
- **Scenario:** Design a user-friendly interface providing current weather information, short-term forecast, and city selection.



# AGILE MODELS



We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- **Individuals** and **interactions** over *processes* and *tools*
- **Working software** over *comprehensive documentation*
- **Customer Collaboration** over *contract negotiation*
- **Responding to change** over *following a plan*

That is, while there is value in the items on the right, we value the items on the left more.

## AGILE PROCESS MODEL

- In 2001, the “Agile Alliance” signed the **Agile Manifesto**. It stated the main values for Agile Development



# AGILE PRINCIPLES

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

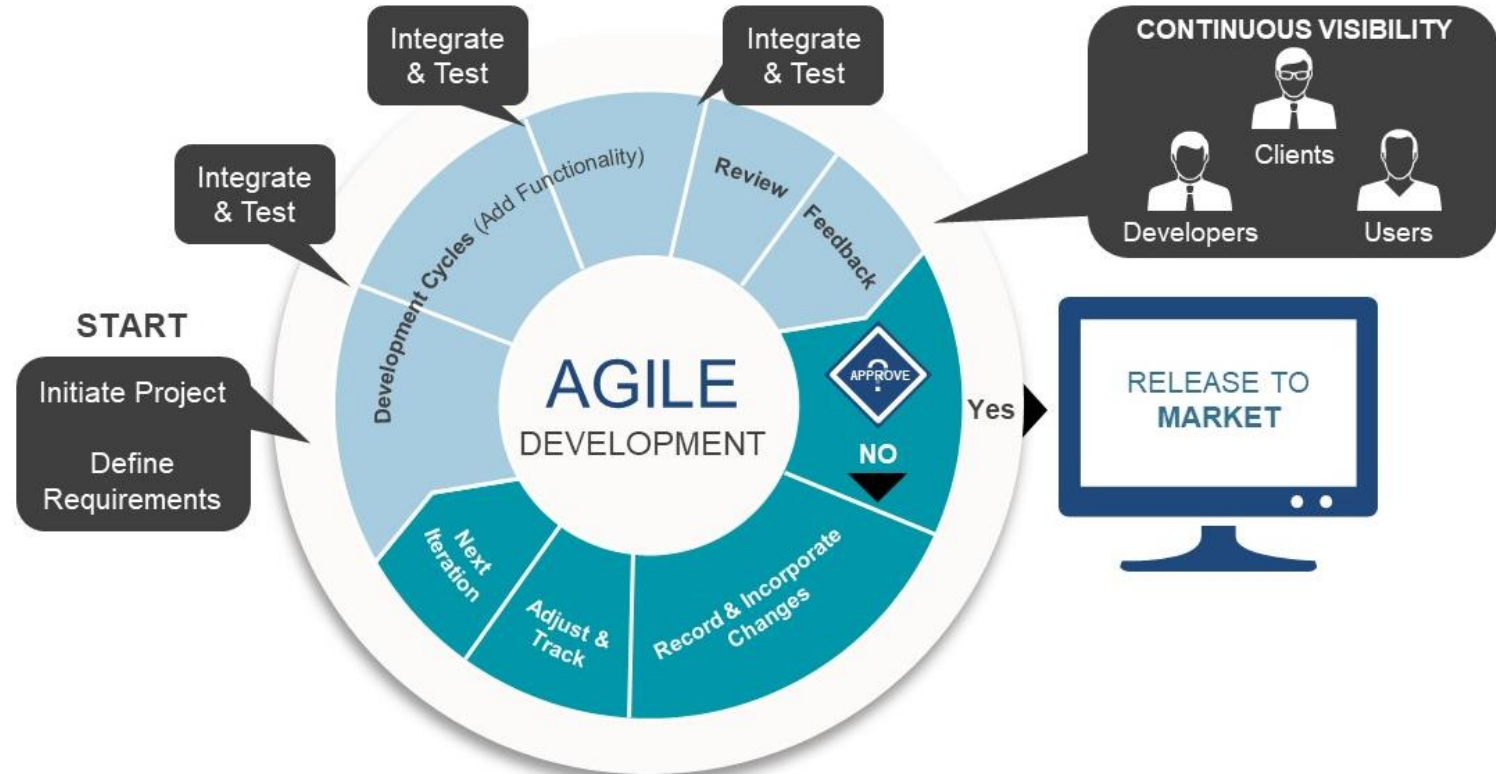
# AGILE PRINCIPLES



6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential. "Do the simplest thing that could possibly work."
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

# EXAMPLES OF AGILE MODELS

- Extreme Programming (XP)
- Scrum





# SCRUM

- Scrum relies on a process pattern called a *sprint*.
- **Sprint**: A time-boxed iteration during which a potentially shippable product increment is created. Sprints are usually 2 to 4 weeks long and provide a consistent rhythm to the development process.
- The number of sprints required will vary depending on *product complexity* and *size*



# SCRUM TEAM

## 1. Product Owner:

The person responsible for defining the features and functionalities of the product, prioritizing the product backlog, and ensuring that the team delivers value to the customer.

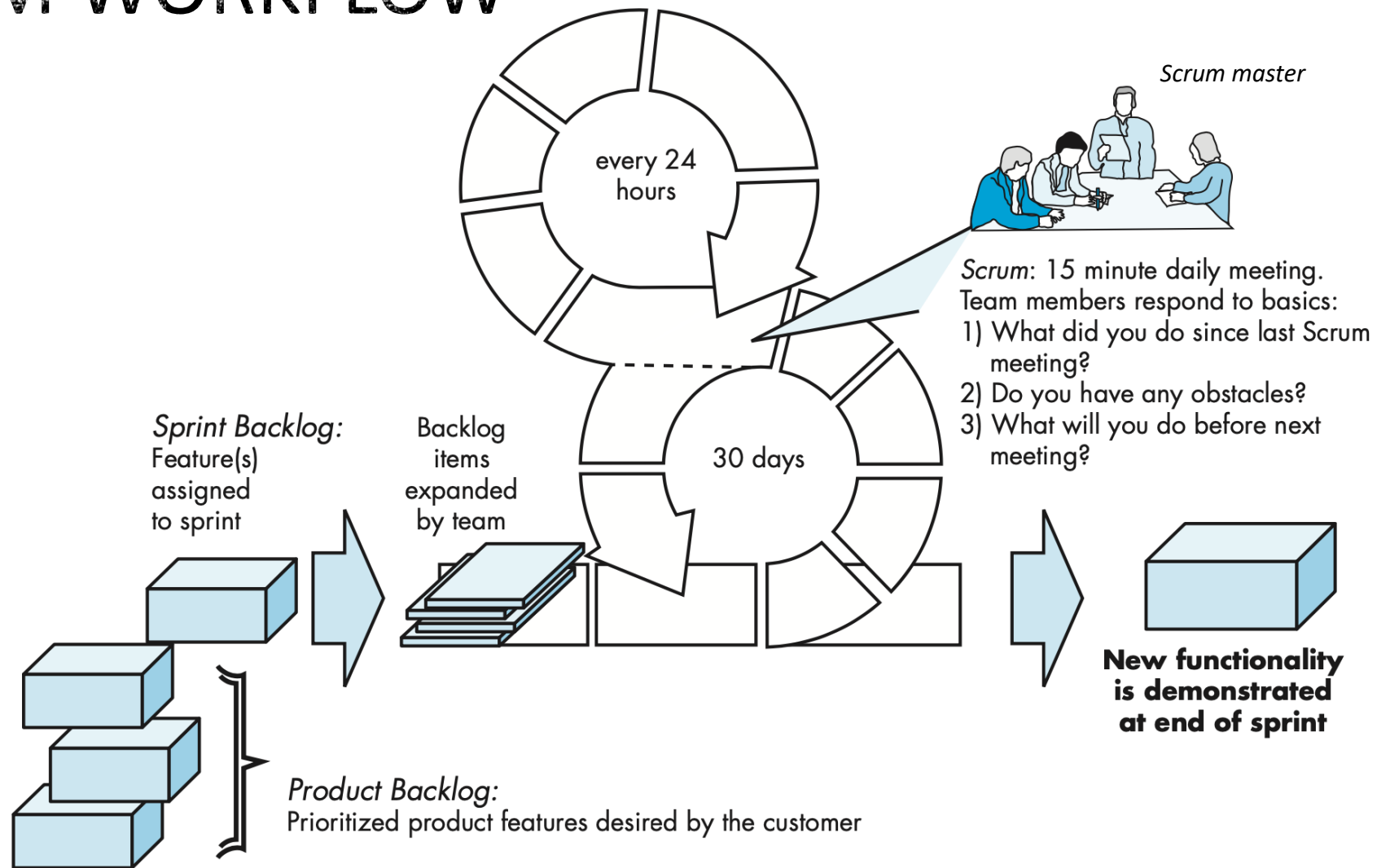
## 2. Scrum Master:

The individual responsible for facilitating the Scrum process, removing impediments, and ensuring that the team follows Scrum practices. The Scrum Master acts as a servant-leader for the team.

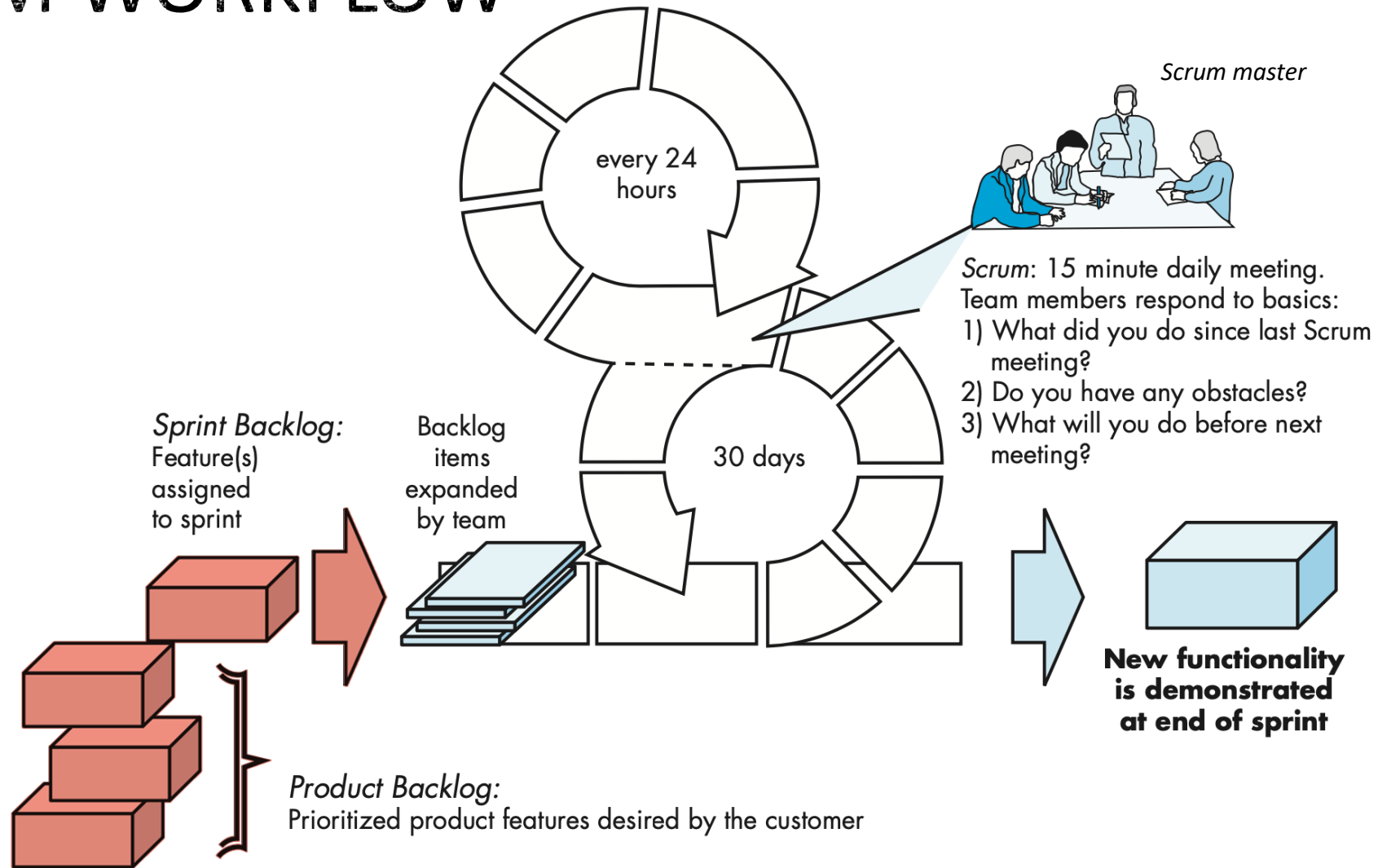
## 3. Development Team:

A self-organizing, cross-functional group of individuals responsible for delivering the product incrementally. The team works collaboratively to achieve the goals set for each sprint. The team will include developers, software testers, software architects.

# SCRUM WORKFLOW



# SCRUM WORKFLOW



# PRODUCT VS. SPRINT BACKLOG

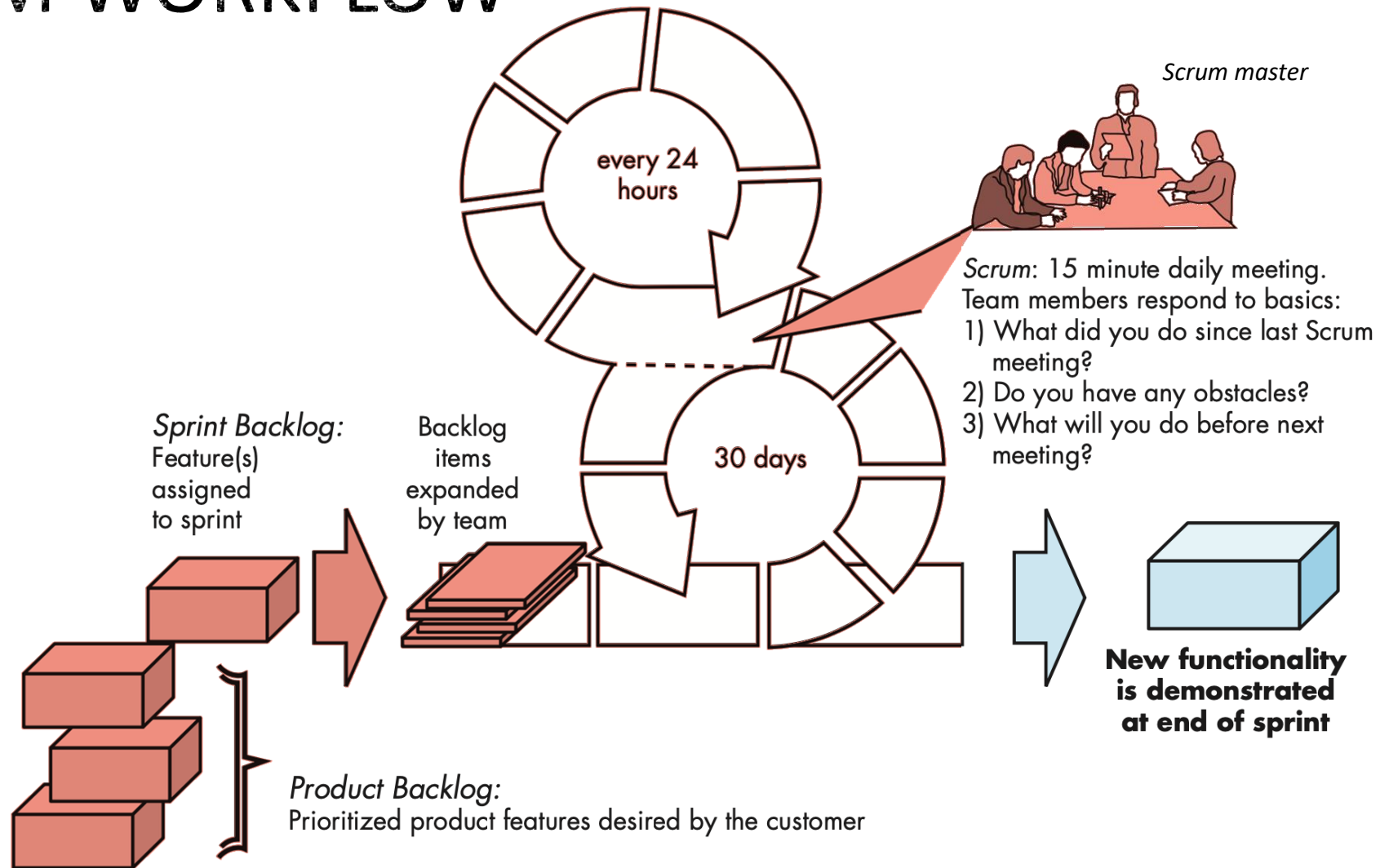
## *Product backlog*

- Prioritized list of all the **product requirements or features** that provide business value for the customer.
- Agreed upon items can be added to the backlog at any time

## *Sprint Backlog:*

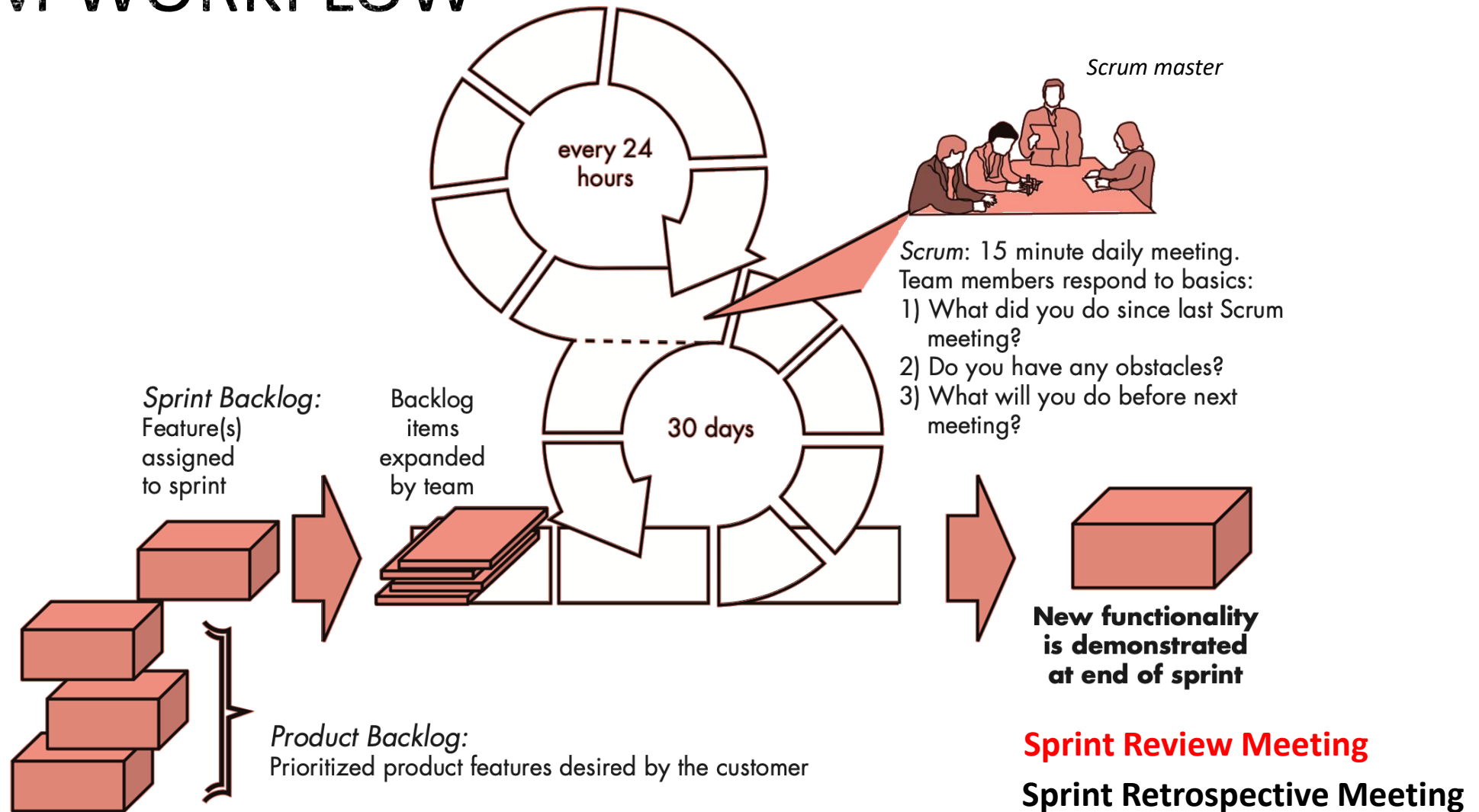
- A subset of product backlog items selected by **the development team** to be completed in the current sprint.
- *No new features can be added to the sprint backlog unless the sprint is cancelled and restarted.*

# SCRUM WORKFLOW












# SCRUM WORKFLOW



# MEETINGS IN SCRUM

- **Sprint Planning:** A meeting at the beginning of each sprint where the team, Product Owner, and Scrum Master collaborate to define the goals for the sprint and select the items from the product backlog to be worked on.
- **Daily Standup (Daily Scrum):** A short, daily meeting where team members discuss their progress, plans for the day, and any impediments they are facing. It helps in keeping everyone informed and the team synchronized.
- **Sprint Retrospective:** A meeting at the end of each sprint where the team reflects on its own processes and identifies opportunities for improvement. It helps in continuous learning and adaptation. It answers questions:
  - What went well and what went wrong in the sprint
  - What could be improved
  - What needs to be done to improve the quality in the next sprint

## Example of a Kanban Board

Backlog	In Progress (3)	Peer Review (3)	In Test (1)	Done	Blocked
					
Fast Track/ Defect					

# READINGS , ASSIGNMENTS, PROJECT TASKS



Read:

- Chapters 2 and 3 of the textbook.



Assignments & Project Tasks:

- The project is open. Start forming the teams and work on Project Deliverable 1.
- Reminder: Project Deliverable 1 is due on Friday 31/1.

A circular inset on the left side of the slide shows a dense pile of 3D question marks. The question marks are dark grey with a metallic or reflective finish, and they are scattered in various orientations, creating a sense of depth and complexity. The background of the slide is a solid dark grey.

Questions?