Week 2 – Software Process Models

CSC 317 / INF 307 – SOFTWARE ENGINEERING

2022/2023 Academic Year

Semester 1

Recommended Reading

- **Chapter 4 & 5** of “Software Engineering: a practitioner’s approach” by Pressman and Maxim

Today

- Software Process Models
  - Plan-driven models
  - Agile models

Recap

- Writing code is very easy
- But engineering good software is HARD

- So how do we make software engineering work?
  - By following a **process**
  - The SDLC is such a generic process

Software Process

- A software development process (or model) is an approach to building, deploying and maintaining software.
- It is a set of sequential activities that take place during the creation of a software system
- Human needs -> requirements -> design -> code -> testing -> installing -> etc.

Software Process

Why do we need processes?

- SE in the real world is chaotic
  - Developer mistakes, changing requirements, etc.
- Allows to identify and repeat good practices
- Helps management know what to do next, when a task is complete, if we are behind schedule, and estimate completion times and costs
- Easy for new team members to know what to do

Software Process

Alternative to having a process

- Not having a defined process is similar to following the **code-and-fix model** which was used in the early days of software development
- Involves two steps: (1) Write some code (2) Fix any problem with the code

- Advantage
  - No overhead since the only activity involved is implementation
- Disadvantages
  - Often matches poorly to customer needs
  - Code deterioration
  - Does not follow solid engineering principles (planning, quality control, etc)

Software Process

- Different software development processes exist
- All processes have some form of the SDLC
  - Can be either rigid, iterative, or even parallel

- Each of these different processes fall somewhere on the continuum below

Software Process

Plan-Driven/Linear

- Waterfall
- Spiral
- Rational Unified Process
- Personal Software Process
- Team Software Process

Agile

- Scrum
- Kanban
- eXtreme Programming
- Incremental
- Iterative

Software Process

Plan-Driven/Linear	Agile
Waterfall	Scrum
Spiral	Kanban
Rational Unified Process	eXtreme Programming
Personal Software Process	Incremental
Team Software Process	Iterative

The choice of a process is affected by organizational, domain/sector, regulatory factors

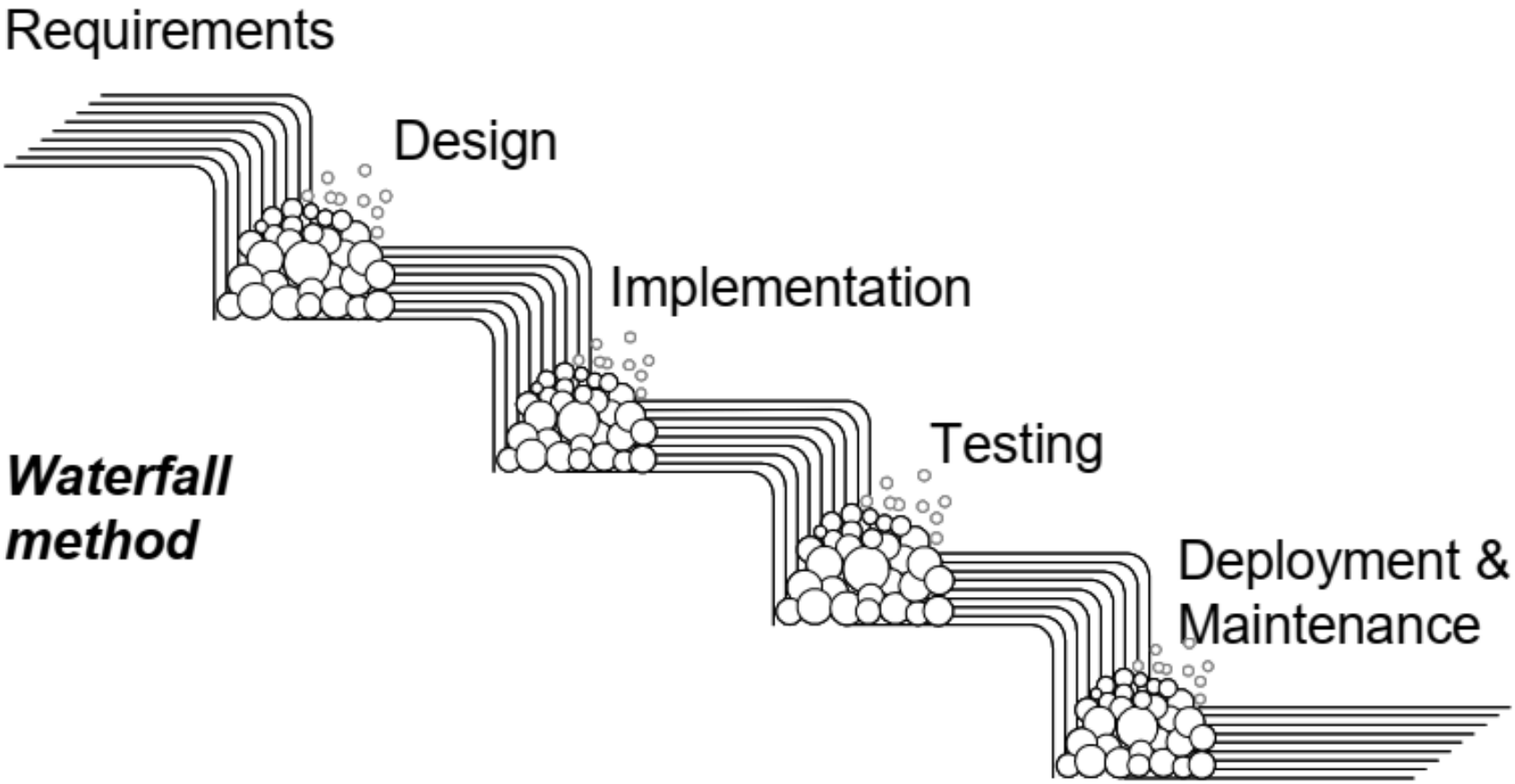
Software Process

- Organizational factors
  - Team structure
  - Location
- Domain/Sector factors
  - E.g., a web application for customer orders? Or are you building a system to monitor the water levels of the Akosombo dam?
- Regulatory factors
  - Requires oversight? E.g., GRA system to integrate e-levy payments with all the financial institutions

Software Process

- By changing the process, we can improve and/or tradeoff:
  - Development speed (time to market)
  - Product quality
  - Project visibility
  - Administrative overhead
  - Risk exposure
  - Customer relations
  - etc

Waterfall model



Waterfall model

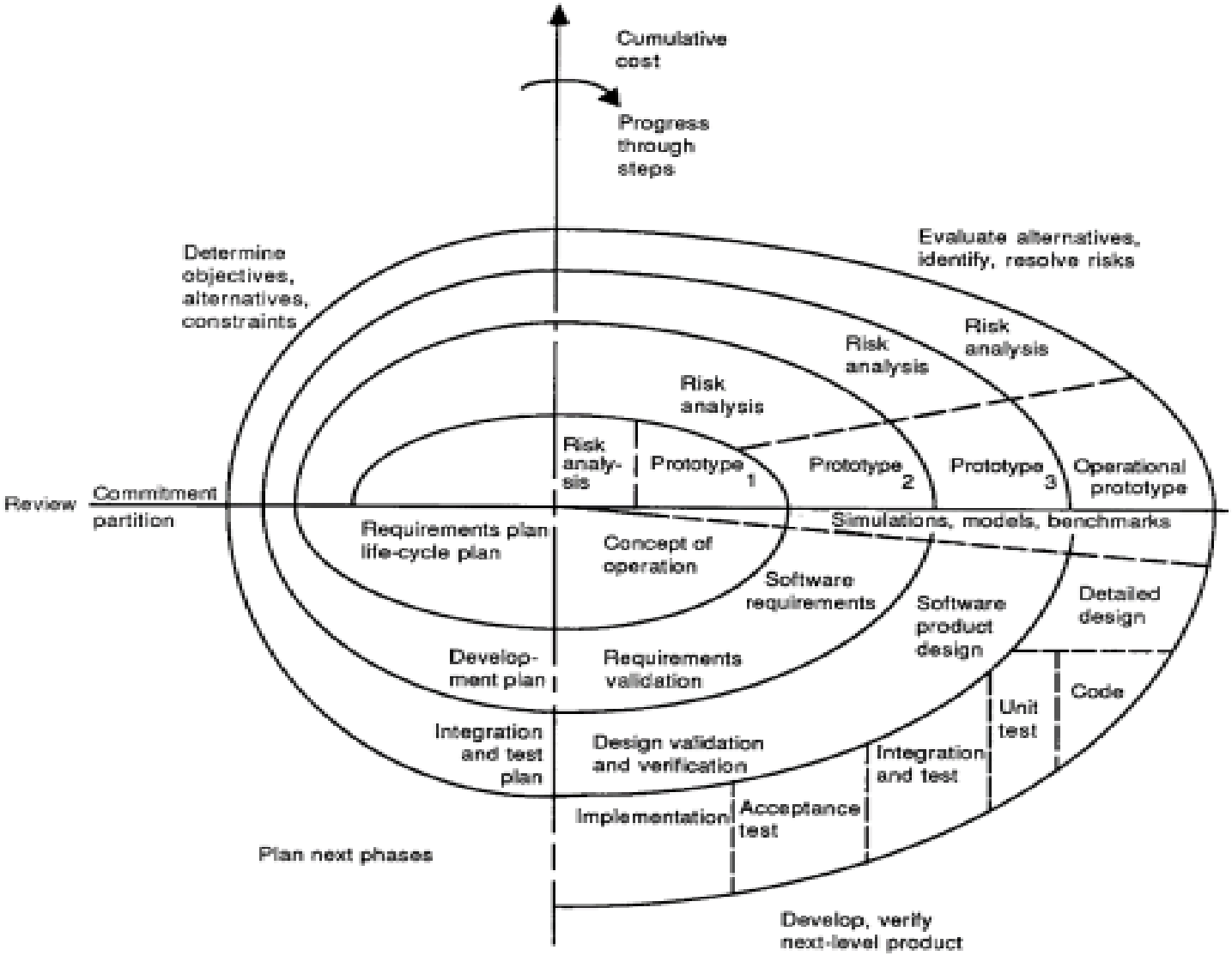
- Follows a linear path of development
- A new phase of the development process is started only once the preceding phase is fully complete
- Very document-driven
  - A document defining what should be done is required at the start of each stage
  - After each stage, a document is also produced (e.g., SRS, code, design, etc)
- Software system is developed as whole, not in increments.
- Discussion. What are the advantages and disadvantages of this model?

Waterfall model (cont.)

- **Advantages**
  - Facilitates strong management control (plan, staff, track).
  - Documents provide a tangible record of accomplishment
  - Easy to understand and use
  - Milestones are well-understood by the team
  - Provides requirements stability

- **Disadvantages**
  - Limited stakeholder feedback or preview of the system
  - Too rigid
    - Requires all requirements to be known upfront
    - Not good for system with changing requirements
  - Little emphasis on prototyping

Spiral model



Spiral model

- Introduced by Boehm (1986) “A Spiral Model of Software Development and Enhancement”
- Consists of several iterations
- Each iteration is a mini-project, solving a sub-problem of the system
- Each iteration follows the waterfall model
  - Includes risk analysis and risk management
  - For each iteration identify and solve the sub-problems with the highest risk.

Spiral model (cont.)

- Each iteration involves:
  - Determining the objectives, constraints and problem addressed by the iteration
  - Generating alternative solutions and identifying (+ resolving) risks
  - Developing and verifying the product
    - Product could be SRS document, high-level design, code, etc
  - Planning for next iteration
  - Review

Spiral model (cont.)

- **Advantages**
  - Integrates risk management in the process
  - Each iteration has a purpose
  - Each iteration can produce prototype/artifact
  - Early attention on alternatives e.g., reusing existing software
- **Disadvantages**
  - Interim prototypes may not be deliverables
  - Risk assessment often expensive and lengthy
  - The process and its management are more complex.
  - End of the project may not be known early so the spiral may go on indefinitely
  - Not suitable for small or low risk projects and could be expensive for small projects.

Incremental Vs. Iterative Models

- These fall somewhere between the continuum.
- May be considered more agile than plan-driven.
- Incremental model – the system is built incrementally i.e., a little bit at a time. It **requires the full requirement specification** to be known at the start of the project.
- Iterative model – similar to the incremental model; it builds the system a bit at a time over a number of iterations. However, it does not require the full specification to be known beforehand.

Agile Processes

- Agile processes evolved from the core planned processes due to :
  - The internet boom
  - Need for different (not so critical) software systems
  - Rush to market (business-oriented)

- Most wanted to be the first to bring their product to market. Having all the requirements in place was no longer a priority.

- The concept of Agile development was officially created in 2001

The Agile Manifesto

http://agilemanifesto.org/

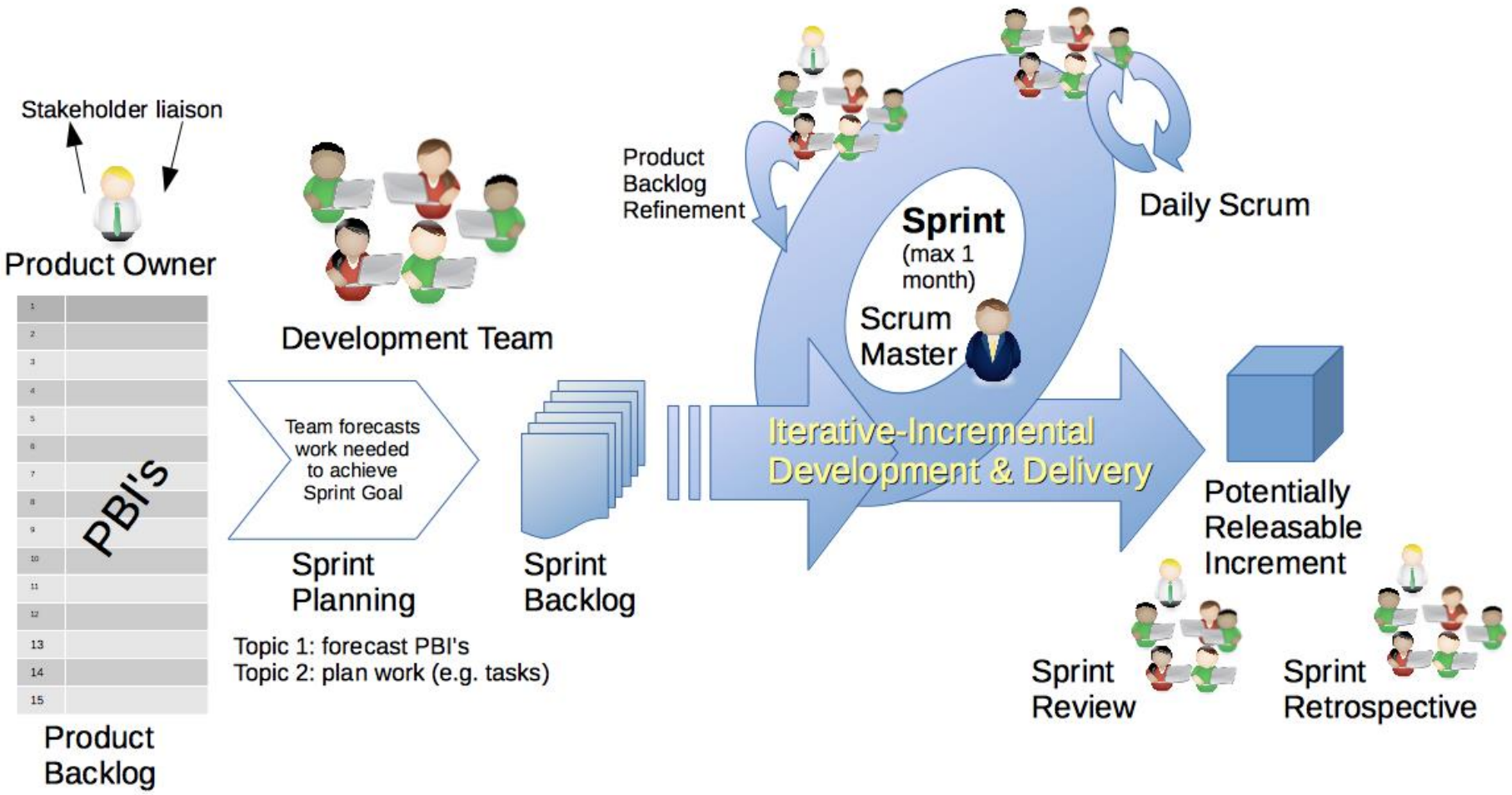
- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Agile processes place more value on items on the left

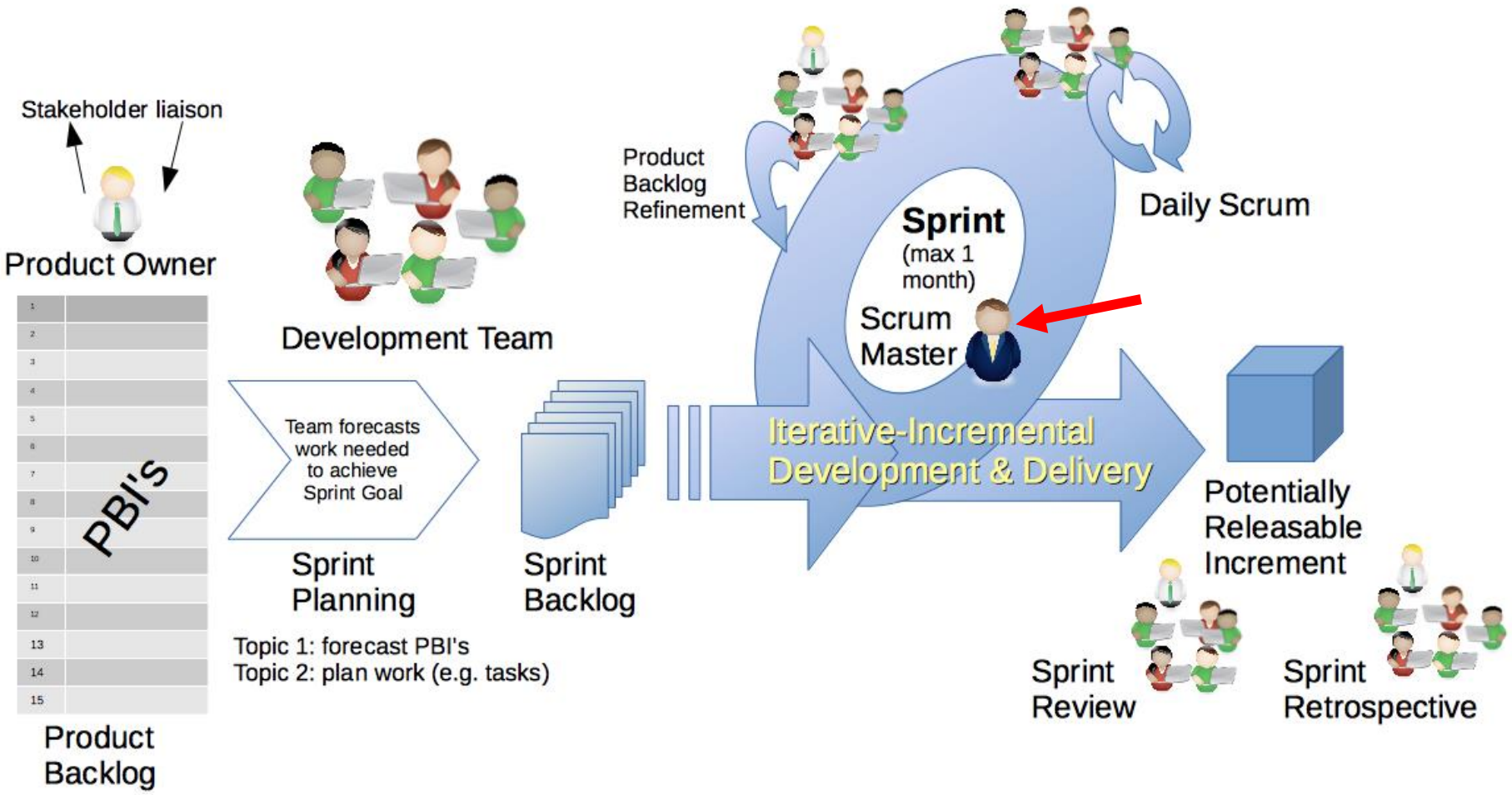
Scrum

- Scrum is the most popular Agile process currently
- Focuses on:
  - Simplicity
  - Constant feedback (from customer)
  - Sprint iterations that always end with a potential shippable product
  - Works on any complex project

Scrum – Overview

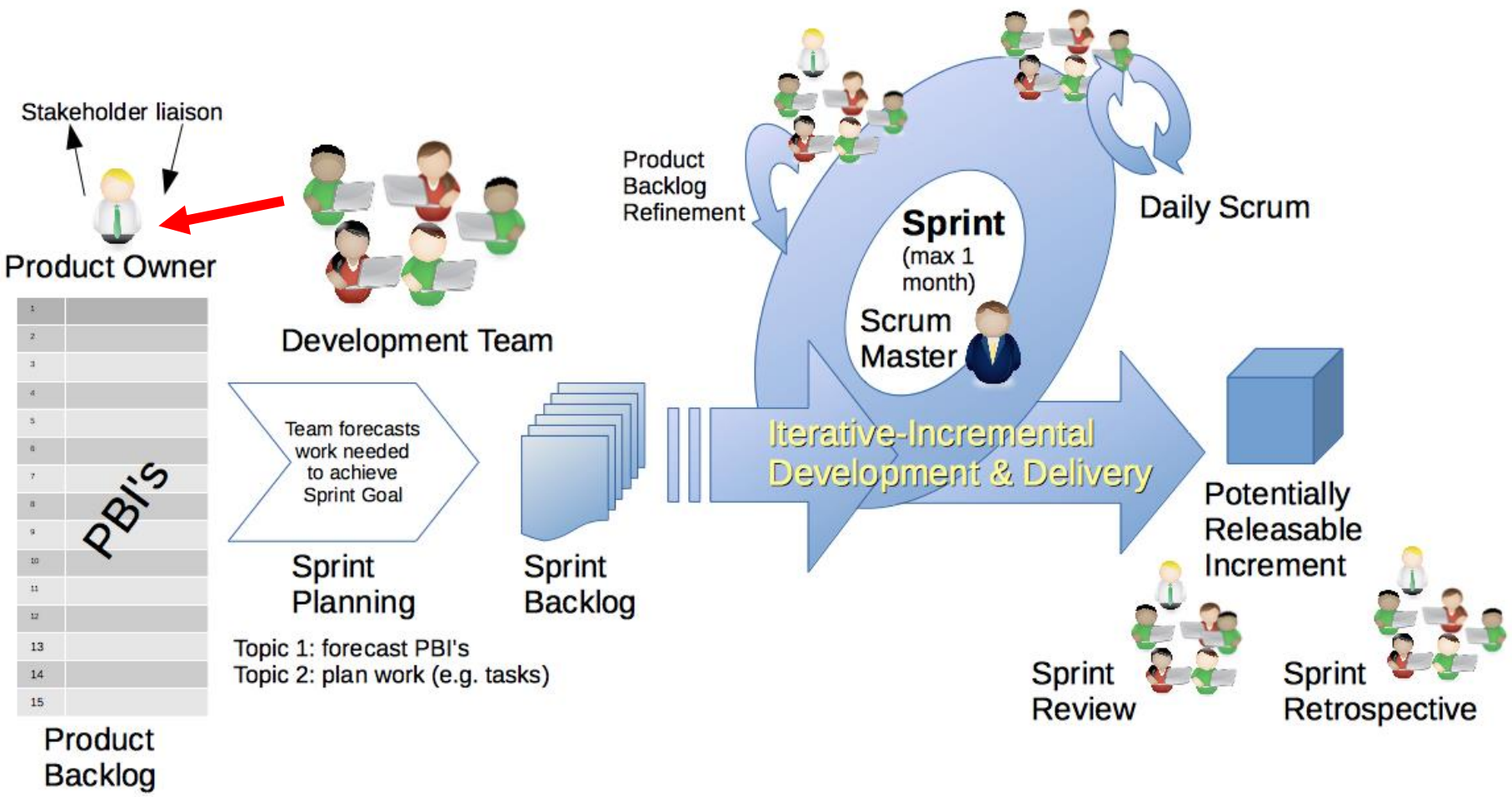


Scrum – Overview



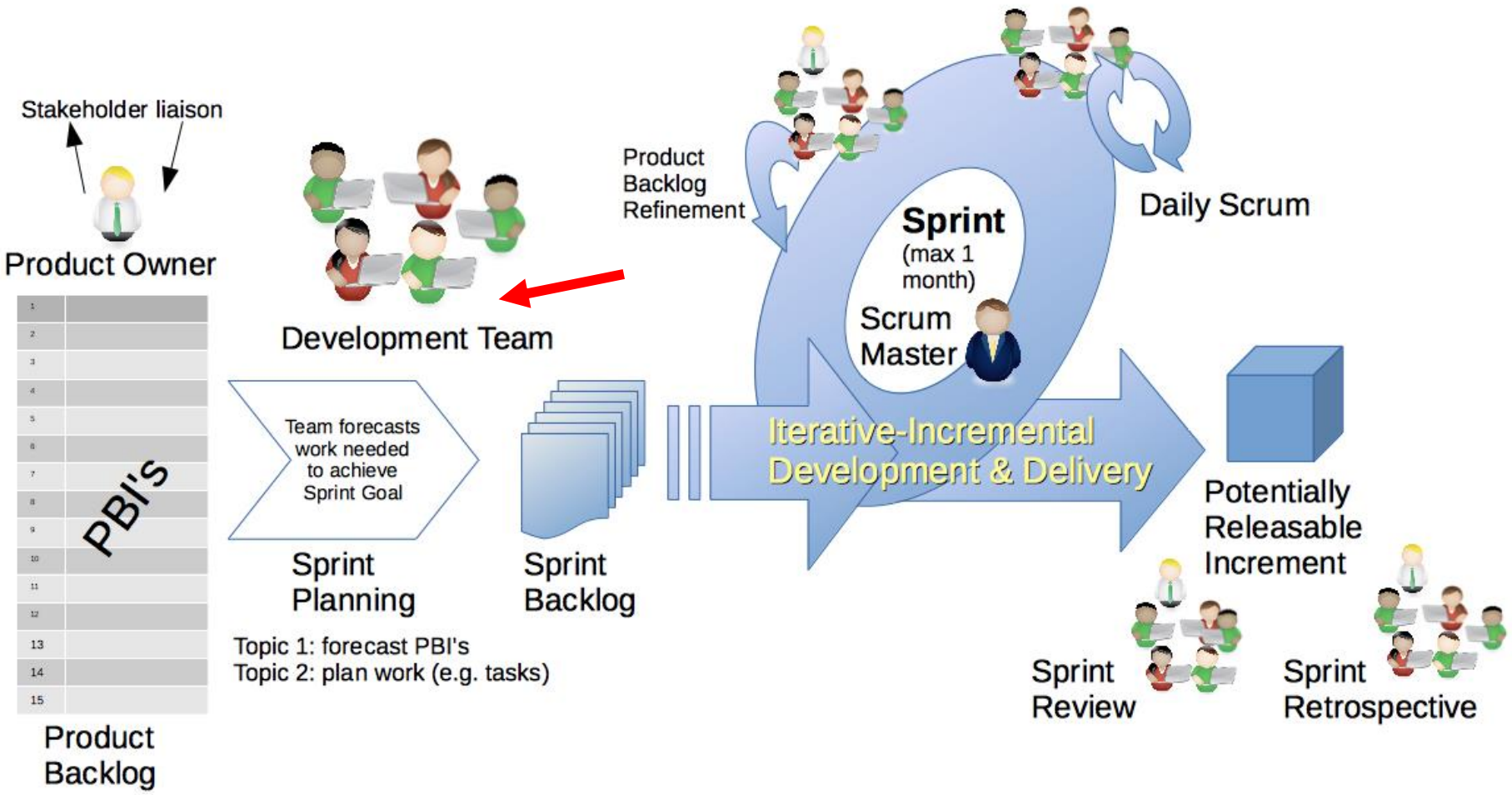
The Scrum Master is responsible for mentoring, coordinating, and facilitating the activities within the team

Scrum – Overview



The Product Owner ensures that the activities of the scrum team align with the bigger picture of the product i.e., meeting customer expectations and market trends

Scrum – Overview



The role of developers spans across architecting and designing the system, programming, testing, etc

Scrum – Overview

- The product owner keeps a **product backlog**, a prioritized "wish list" of all requirements and features
- Development in this model works in increments of time called **Sprints**
  - Usually 2 or 4 weeks
- At the beginning of each sprint, there is a sprint planning.
  - Some items are moved form the product backlog into the sprint backlog (by the ScrumMaster)
  - These features must be implemented within the set timeframe of the sprint

Scrum – Overview

- **Daily Scrum**: there is a daily meeting to assess the progress of the current sprint. Also called Stand-up meetings

- The sprint ends with a review and retrospective, and the outcome is a potential shippable product
  - The sprint review allows for an artifact (a potential shippable product) of the work done so far to be shared with stakeholders for feedback
  - the sprint retrospective is an internal meeting within the team to learn and improve its process.

Agile models (cont.)

Advantages

- Quick feedback due to daily interaction with customer representative
- Easier to adapt to changing requirements
- Issues detected and fixed faster
- Less bureaucracy and documentation
- More room to experiment and test ideas

Disadvantages

- Documentation gets sidetracked
- No clear scope of project; project may become ever-lasting
- Technical debt
- Difficult to estimate cost and required effort

Discussion

- Which process model is best suited for the following?
  - a software system that manages the autopilot feature of an airplane
  - a management system for a restaurant

References

- Pressman, R. S. and Maxim, B. R. (2014). Software Engineering: a practitioner's approach, 8th edition: McGraw-Hill
- Winston R. Royce (1970). "Managing the Development of Large Software Systems", in Proc. of IEEE WESCON, pp. 1-9.
- B. W. Boehm (1988). A Spiral Model of Development and Enhancement, in IEEE Computer, pp. 61-72