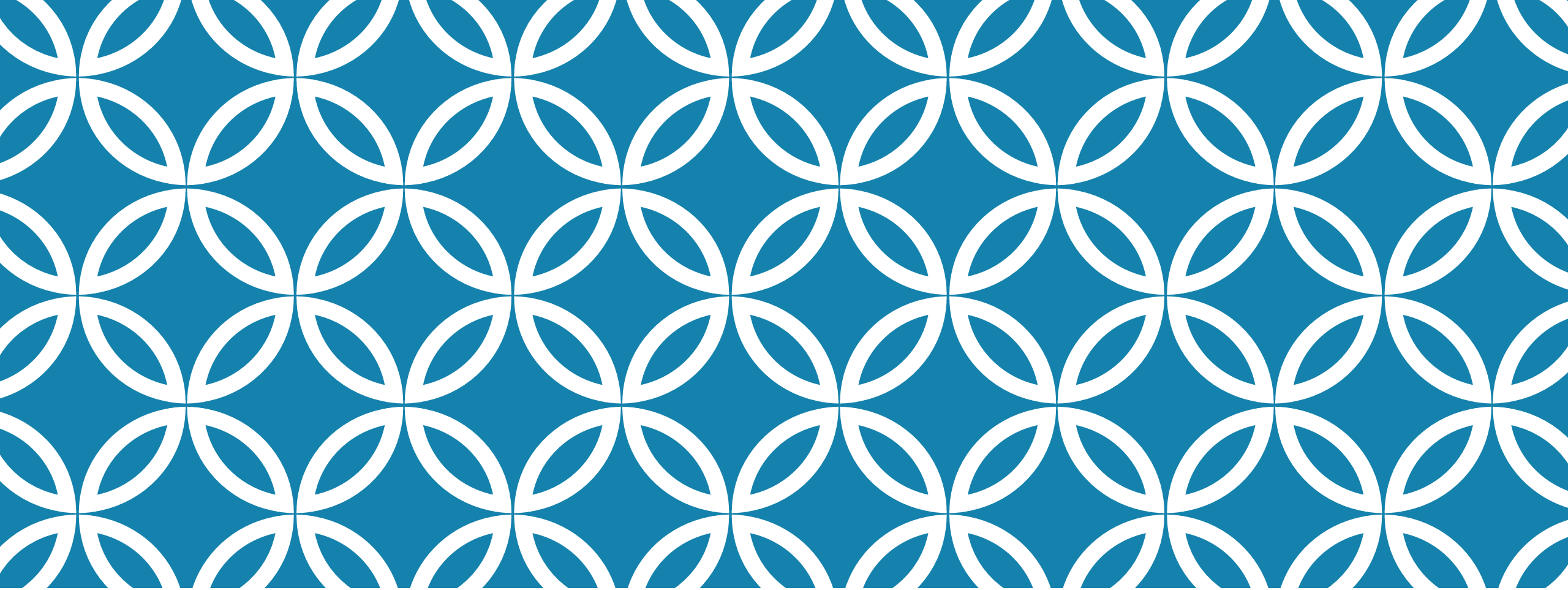


SOFTWARE PROCESS REVIEW

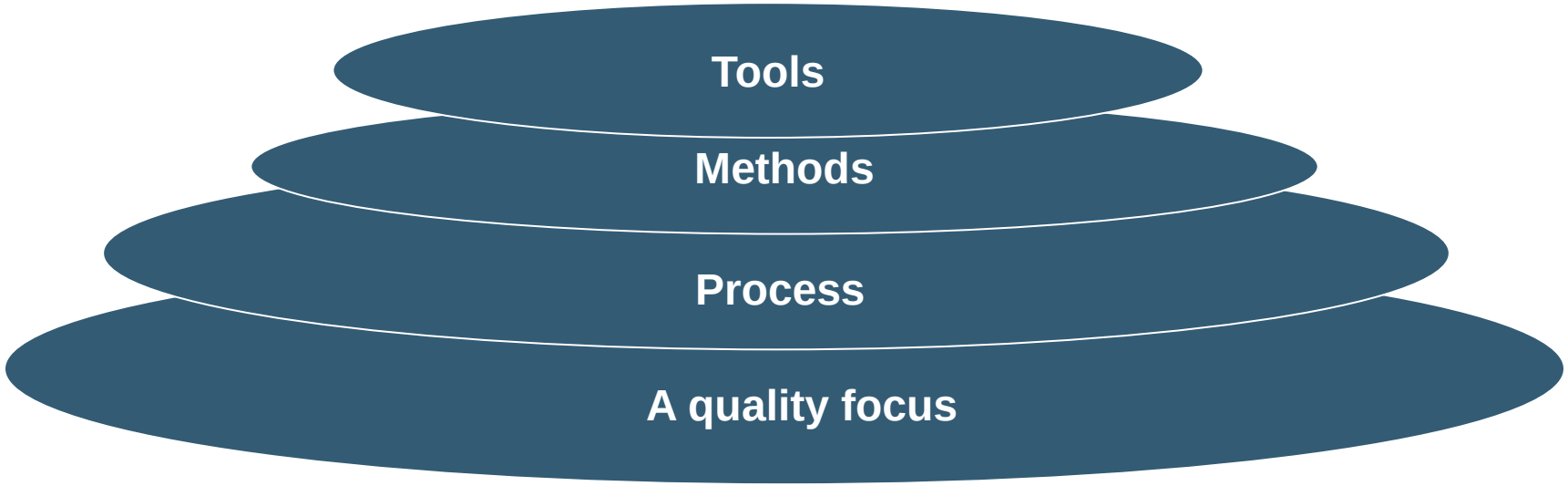
IF3250 PROYEK PERANGKAT LUNAK

Sem II 2024/2025

SOFTWARE ENGINEERING

- Software engineering is the establishment of sound **engineering principles** in order to obtain **reliable** and **efficient** software in an **economical** manner.
- Software engineering is the application of a **systematic, disciplined, quantifiable** approach to the **development, operation,** and **maintenance** of software.
- Software engineering encompasses a **process, management techniques, technical methods,** and the **use** of tools.

SOFTWARE ENGINEERING – A LAYERED TECHNOLOGY



GENERIC SOFTWARE PROCESS FRAMEWORK

Communication

- System analyst vs User
- System analyst vs Programmer

Planning

- Cost, Time, human resources, etc

Modeling

- Structured approach
- Object oriented approach

Construction

- Coding and Testing

Deployment

- Software delivery to customer

UMBRELLA ACTIVITIES

Software project tracking and control

- allows the software team to **assess progress** against the project plan and **take** any necessary **action** to maintain the schedule.

Risk management

- **assesses risks** that may affect the **outcome** of the project or the **quality** of the product.

Software quality assurance

- defines and conducts the activities required to **ensure** software **quality.**

Technical reviews

- assesses software engineering work products in an effort to **uncover** and **remove errors** before they are propagated to the next activity.

UMBRELLA ACTIVITIES

Measurement

- defines and collects **process, project,** and **product measures** that assist the team in delivering software that meets **stakeholders' needs;** can be used in conjunction with all other framework and umbrella activities.

Software configuration management

- manages the **effects** of **change** throughout the software process.

Reusability management

- defines **criteria** for work product **reuse** (including software components) and establishes mechanisms to **achieve reusable** components.

Work product preparation and production

- encompasses the activities required to create work products such as **models, documents, logs, forms,** and **lists.**

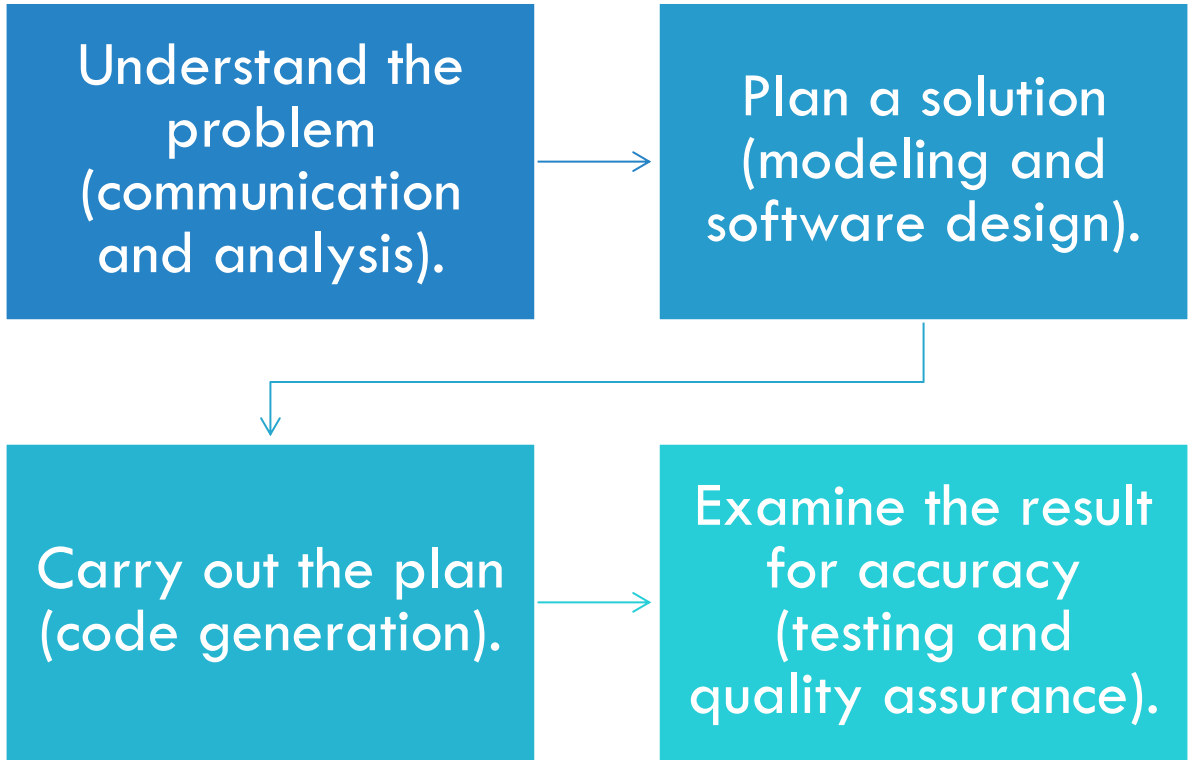
PROCESS ADAPTATION

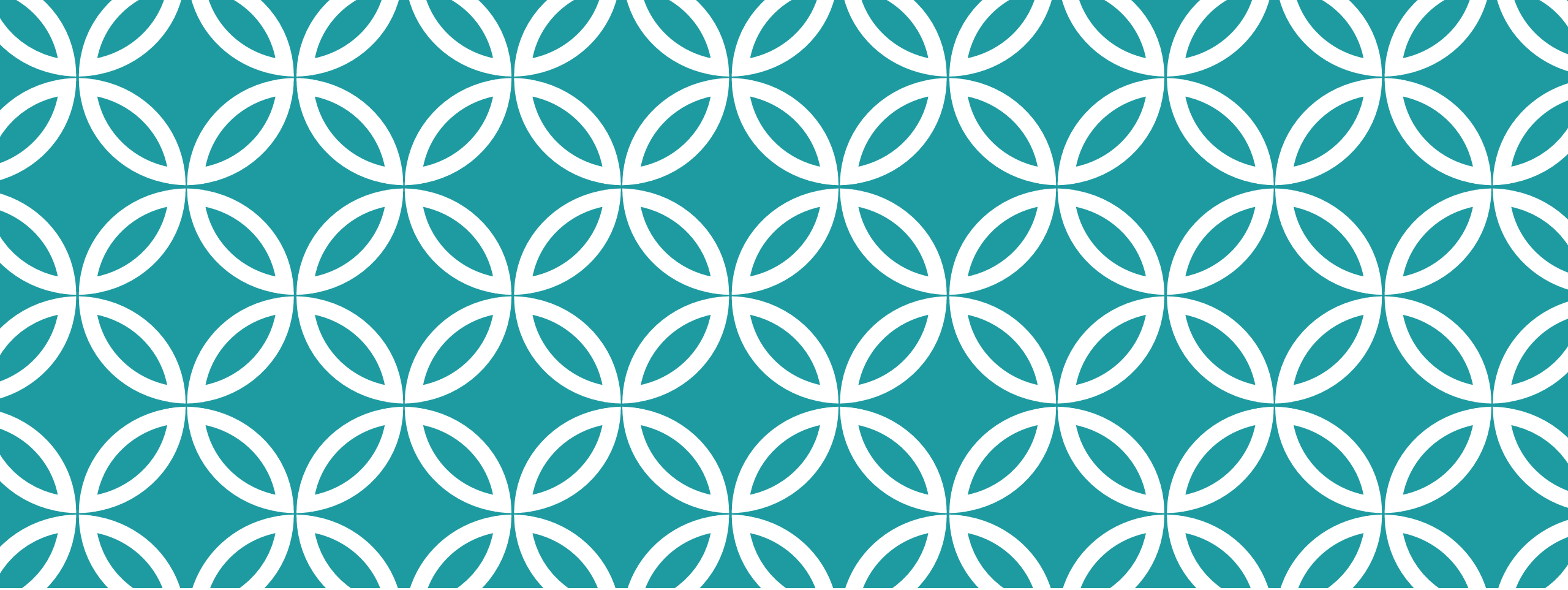
The software engineering process should be agile and adaptable

- to the problem,
- to the project,
- to the team, and
- to the organizational culture

A process adopted for one project might be significantly different than a process adopted for another project.

THE ESSENCE OF SOFTWARE ENGINEERING PRACTICE



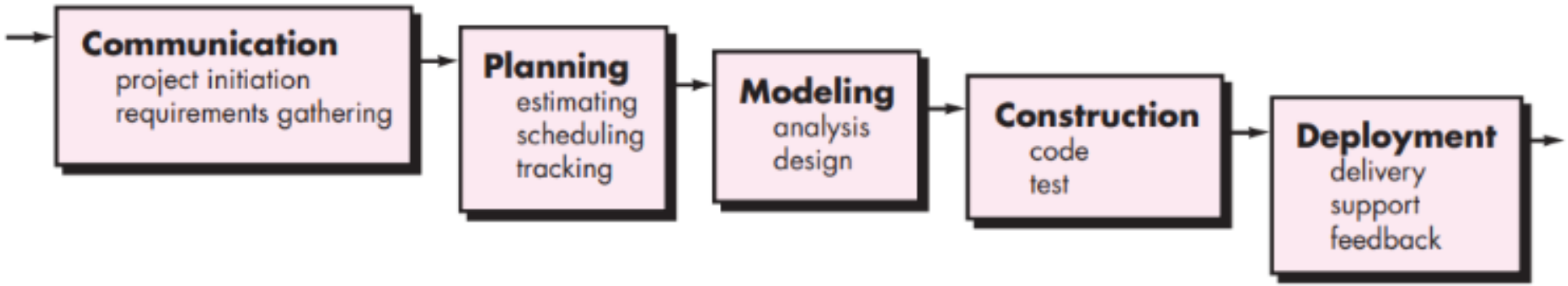


PROCESS MODELS

- Prescriptive Process Models
- Specialized Process Models
- Unified Process
- Agile Software Development

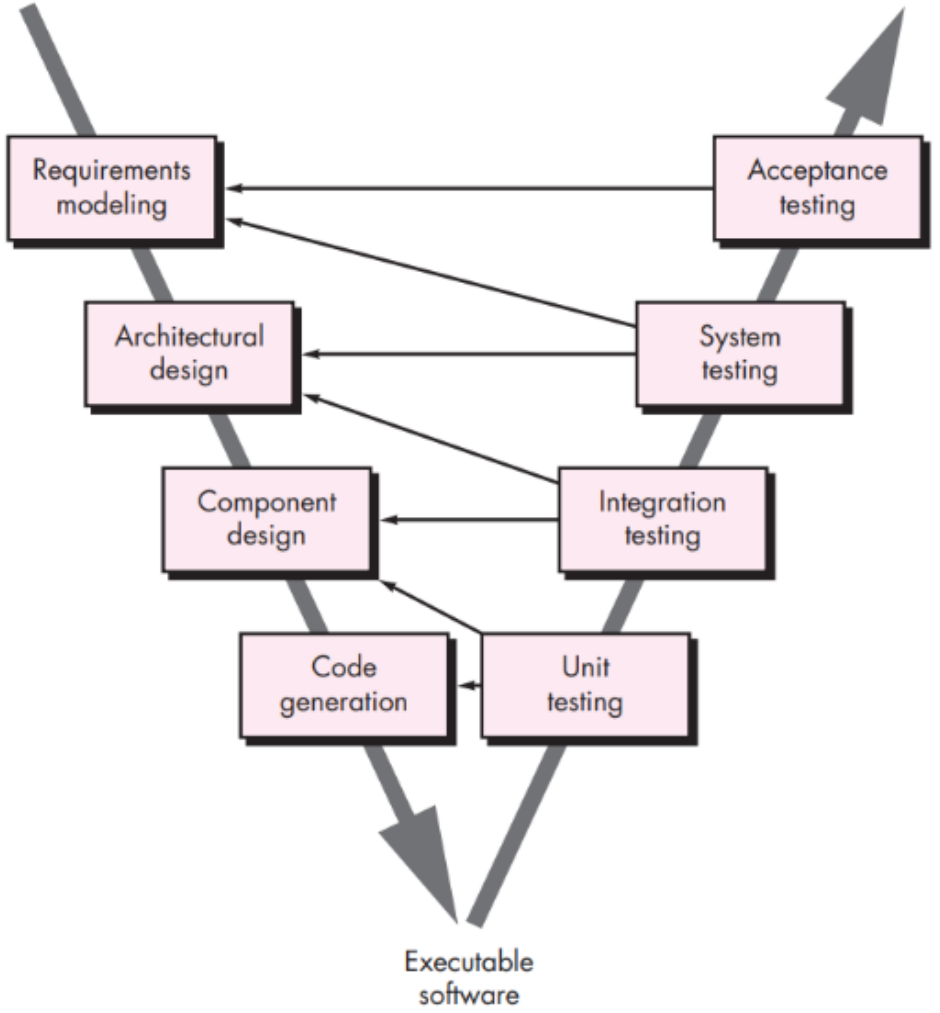
PRESCRIPTIVE PROCESS MODELS

The Waterfall Model - *classic life cycle*



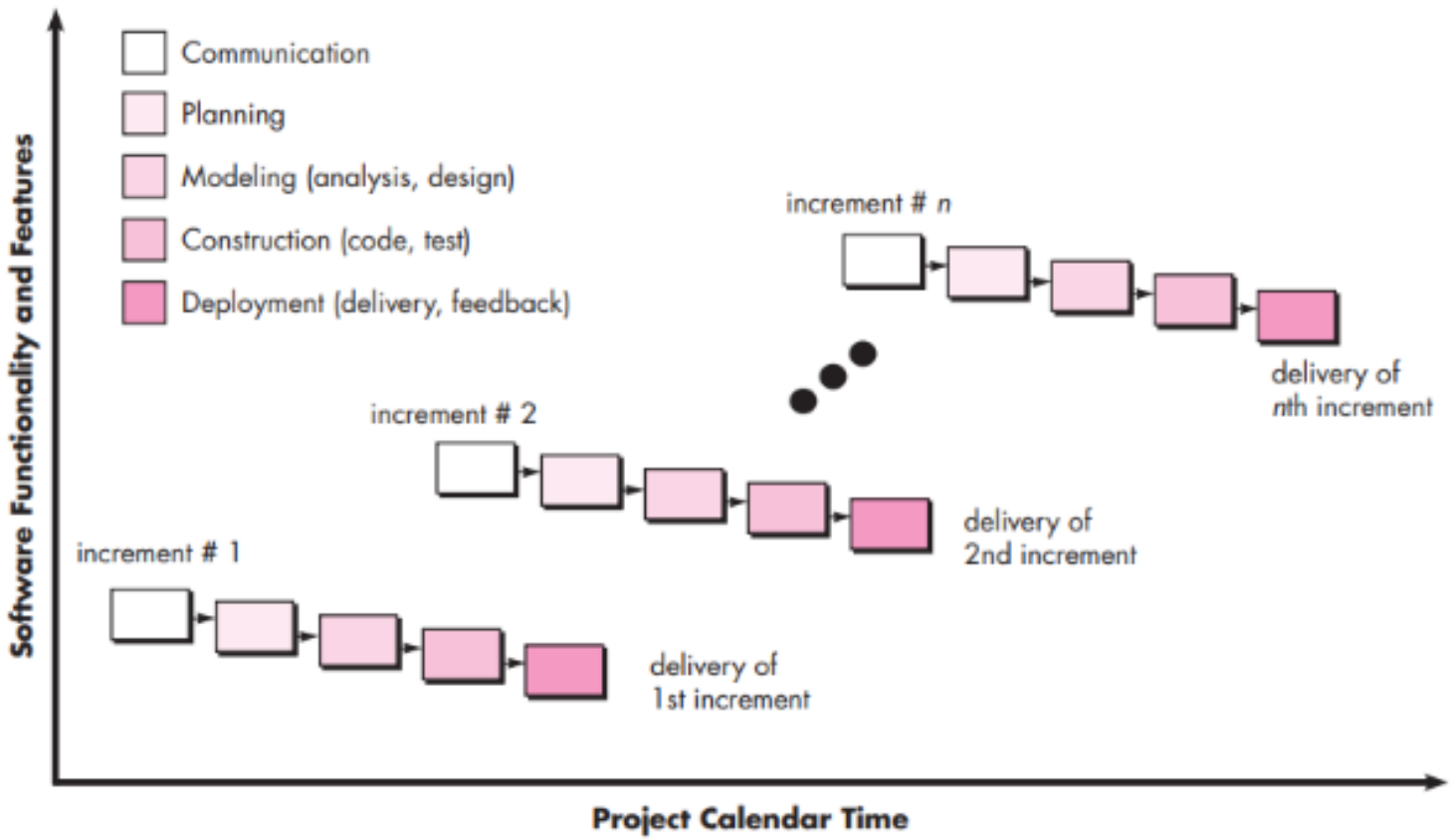
PRESCRIPTIVE PROCESS MODELS (2)

The V model



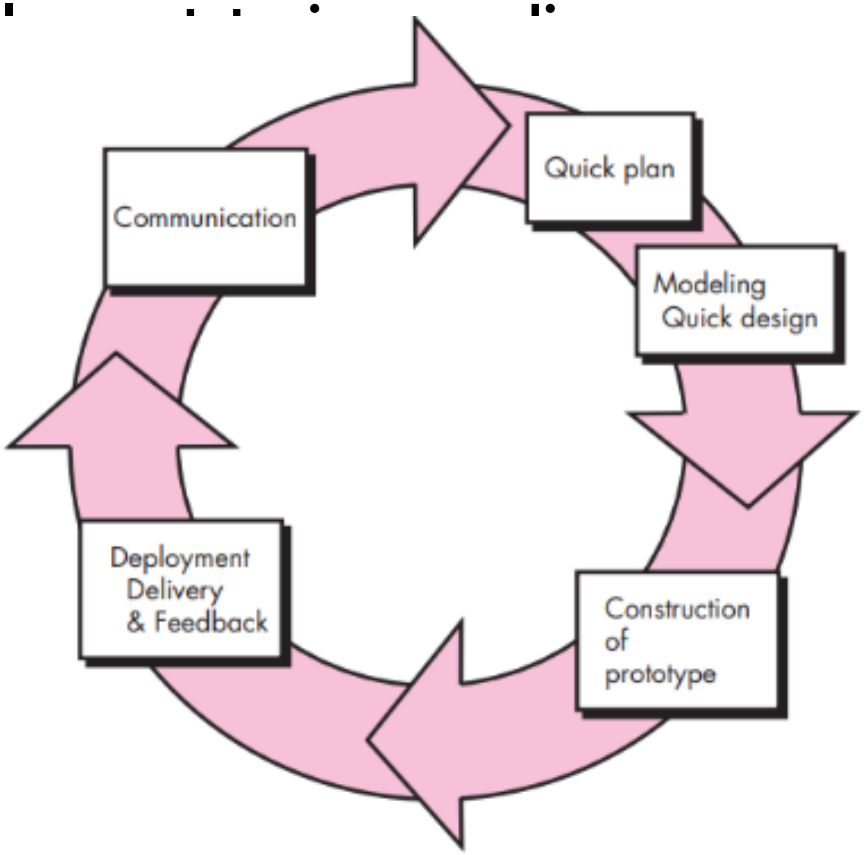
PRESCRIPTIVE PROCESS MODELS (3)

Incremental Process Models



PRESCRIPTIVE PROCESS MODELS (4)

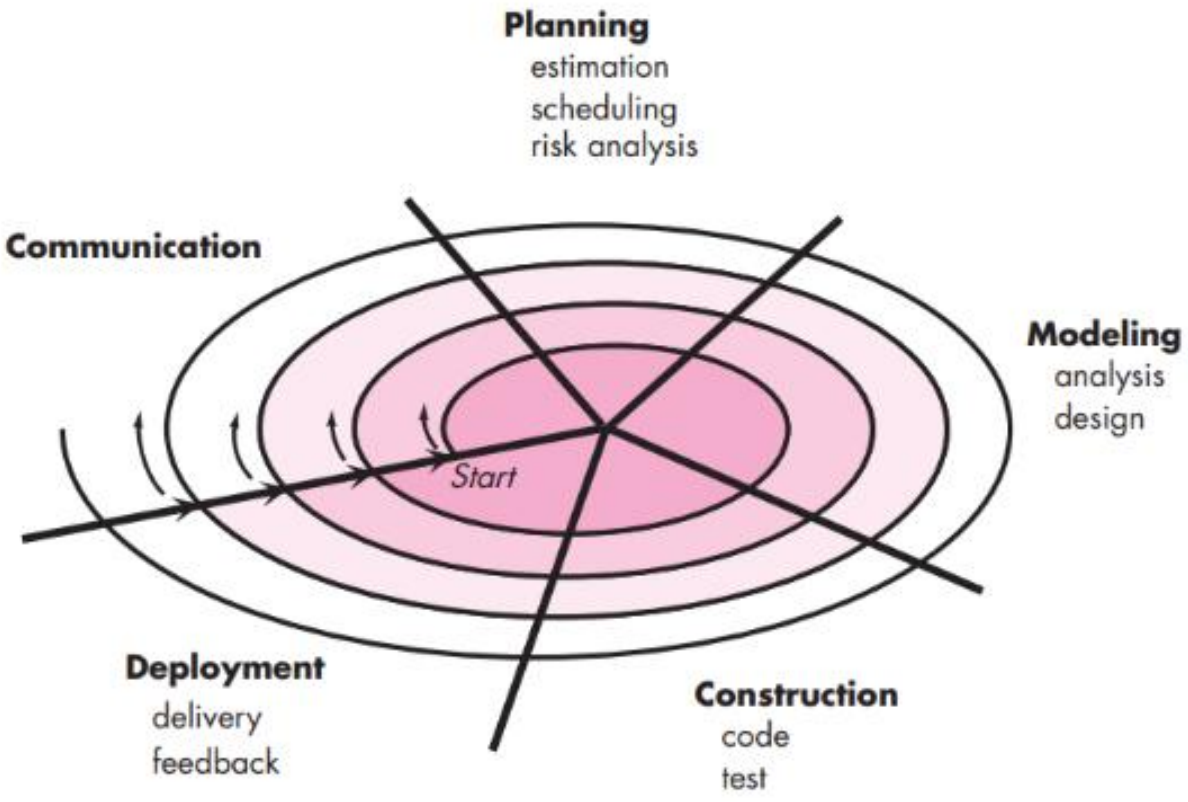
Evolutionary Process Mo



PRESCRIPTIVE PROCESS MODELS (5)

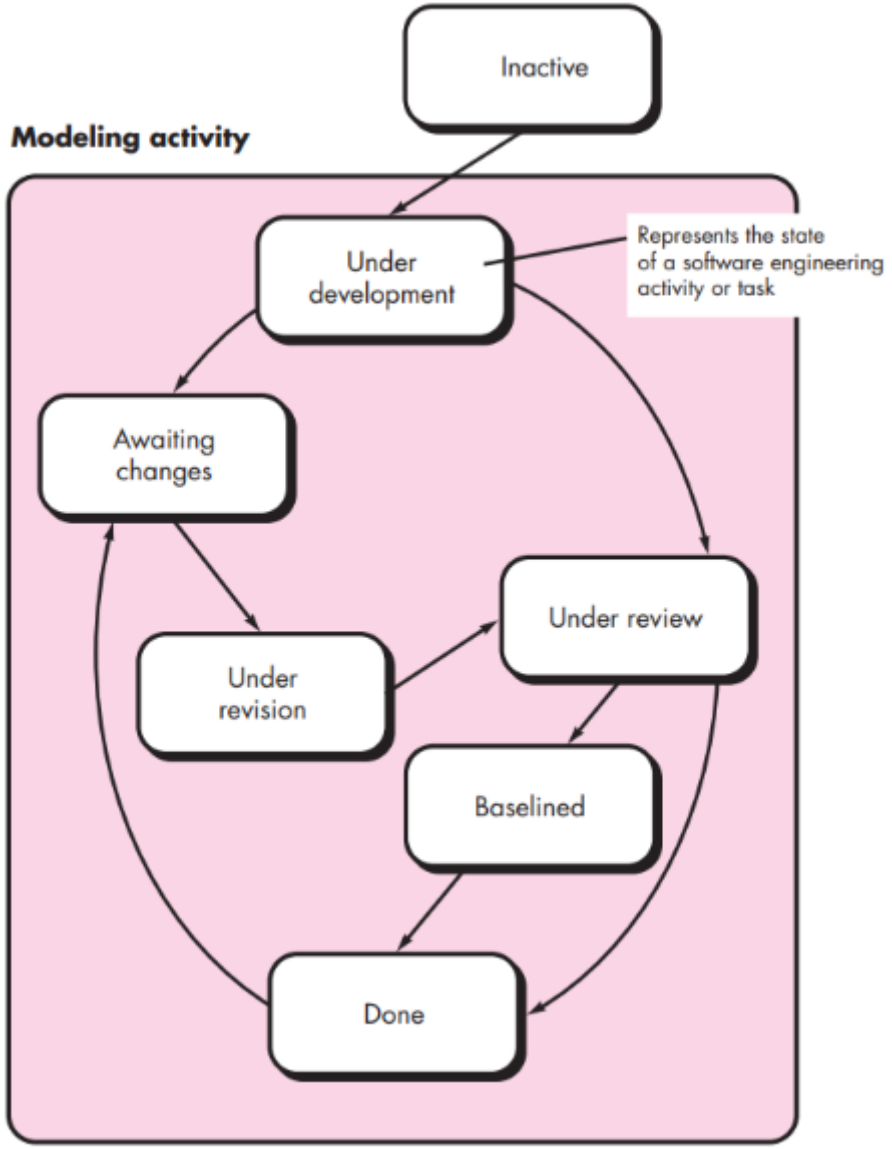
Evolutionary Process Models –

the Spiral Model



PRESCRIPTIVE PROCESS MODELS (6)

Concurrent Models



SPECIALIZED PROCESS MODELS

Component-Based Development - comprises applications from prepackaged software components.

The Formal Methods Model

- encompasses a set of activities that leads to **formal mathematical specification** of computer software, enable you to specify, develop, and verify a computer-based system by applying a rigorous, mathematical notation.
- the formal methods approach has gained adherents among software developers who must **build safety-critical software** (e.g., developers of aircraft avionics and medical devices) and among developers that would **suffer severe economic hardship should software errors occur**

Aspect-Oriented Software Development

- paradigm that provides a process and methodological approach for defining, specifying, designing, and constructing *aspects* — "*mechanisms beyond subroutines and inheritance for* localizing the expression of a crosscutting concern"

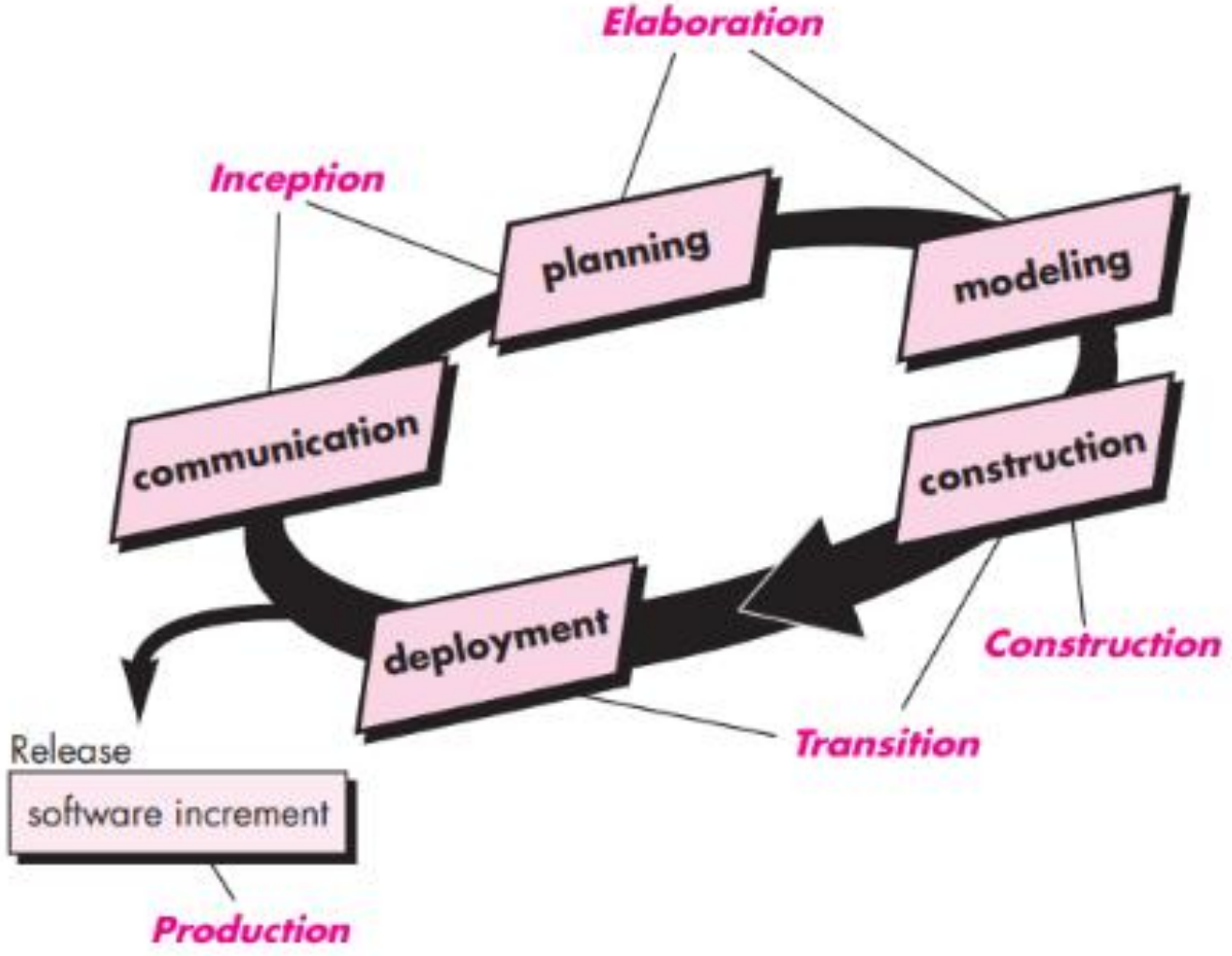
UNIFIED PROCESS

A “use case driven, architecture-centric, iterative and incremental” process model

The result was UML models —a *unified modeling*

language that contains a robust notation for the modeling and development of object- oriented systems.

UNIFIED PROCESS (2)



AGILE SOFTWARE DEVELOPMENT METHODOLOGY

Agile Methodology is develop based on iterative and incremental development

It is based on feedback from the clients.

In this methodology the requirements and solutions evolve through collaboration between self-organizing, cross-functional teams who work in close liaison with the clients.

It promotes evolutionary development, adaptive planning and encourages rapid and flexible response to change.

It is suitable for

- Product (and less for services)
- Smaller to medium sized products

Development teams for such products is ranging from 5 to 20-20 members

- A team of 100 members was also noted to be a success

GOALS & POTENTIAL BENEFITS OF AGILITY

- Delivering the most value to the business, efficient use of resources, maximize ROI and time-to-ROI
- Faster development, higher productivity
- Flexibility to respond to change and leverage learning
- Better quality solutions, more enduring systems
- More fulfilling development culture

AGILE MANIFESTO

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 value in the items on the right, we value the items on the left more."

CORE PRINCIPLES

- “Fits just right” process
- Continuous testing and validation
- Consistent team collaboration
- Rapid response to change
- Ongoing customer involvement
- Frequent delivery of working software

AGILE METHODOLOGIES

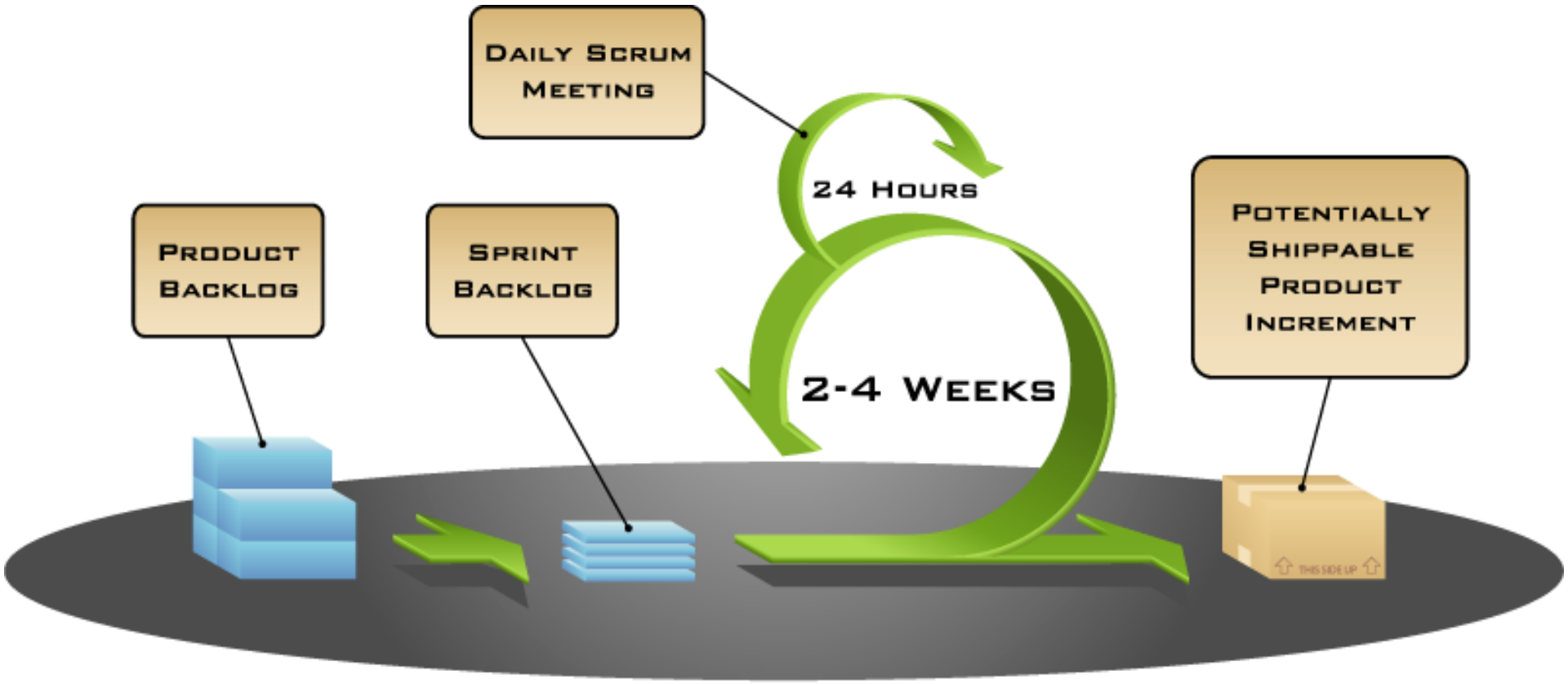
- Adaptive Software Development
- Agile Modeling
- Agile Unified Process (AUP)
- Crystal
- Dynamic Systems Development Method (DSDM)
- Essential Unified Process (EssUP)
- Extreme Programming (XP)
- Feature Driven Development (FDD)
- Lean Development
- Open Unified Process (OpenUP)
- **Scrum**
- Velocity tracking

SCRUM

CHARACTERISTICS

- Self-organizing teams
- Product progresses in a series of month-long “sprints”
- Requirements are captured as items in a list of “product backlog”
- No specific engineering practices prescribed
- Uses generative rules to create an agile environment for delivering projects
- One of the “agile processes”

HOW IT WORKS


Daily Scrum Meeting
24 Hours
Product Backlog
Sprint Backlog
2-4 Weeks
Potentially Shippable Product Increment
Copyright © 2005, Mountain Goat Software

SPRINTS

Scrum projects make progress in a series of “sprints”

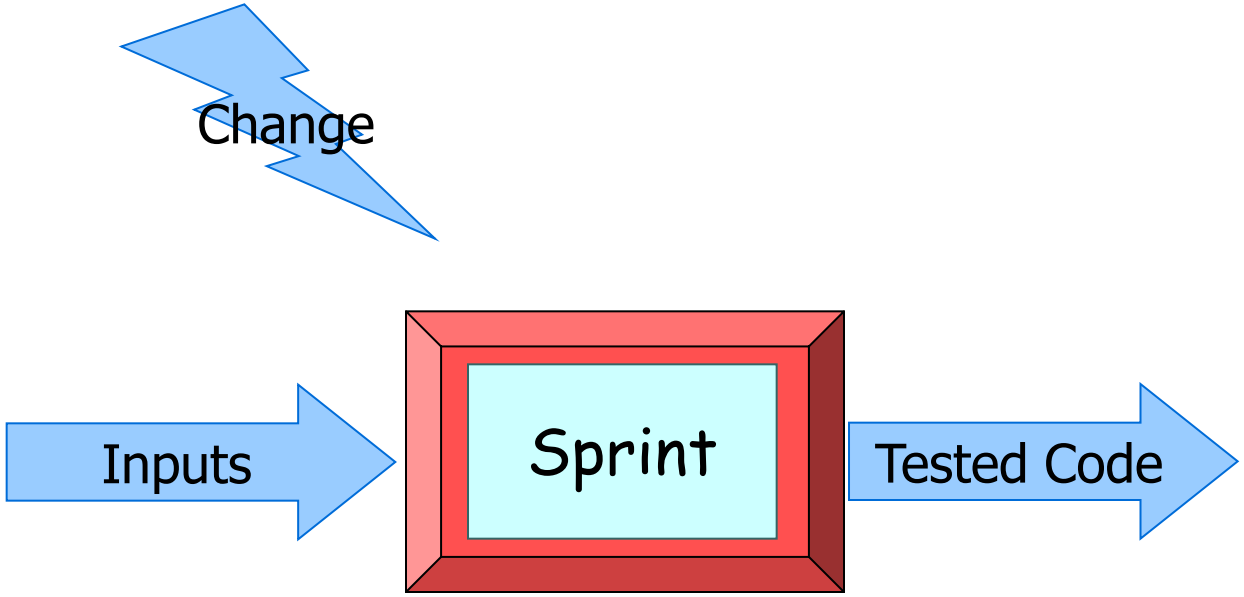
Target duration is one month

+/- a week or two

- But, a constant duration leads to a better rhythm

Product is designed, coded, and tested during the sprint

NO CHANGES DURING THE SPRINT



Plan sprint durations around how long you can commit to keeping change out of the sprint

SCRUM FRAMEWORK

Roles : Product Owner, ScrumMaster, Team

Ceremonies : Sprint Planning, Sprint Review, Sprint Retrospective, & Daily Scrum Meeting

Artifacts : Product Backlog, Sprint Backlog, and Burndown Chart (+ Kanban Board)

PRODUCT OWNER

1. Define the features of the product
2. Decide on release date and content
3. Be responsible for the profitability of the product (ROI)
4. Prioritize features according to market value
5. Adjust features and priority every iteration, as needed
6. Accept or reject work results.

THE SCRUM MASTER

Represents management to the project

Responsible for enacting Scrum values and practices

Removes impediments

Ensure that the team is fully functional and productive

Enable close cooperation across all roles and functions

Shield the team from external interferences

SCRUM TEAM

Typically 5-10 people

Cross-functional

- QA, Programmers, UI Designers, etc.

Members should be full-time

- May be exceptions (e.g., System Admin, etc.)

Teams are self-organizing

- What to do if a team self-organizes someone off the team??
- Ideally, no titles but rarely a possibility

Membership can change only between sprints

CEREMONIES

SPRINT PLANNING MEETING

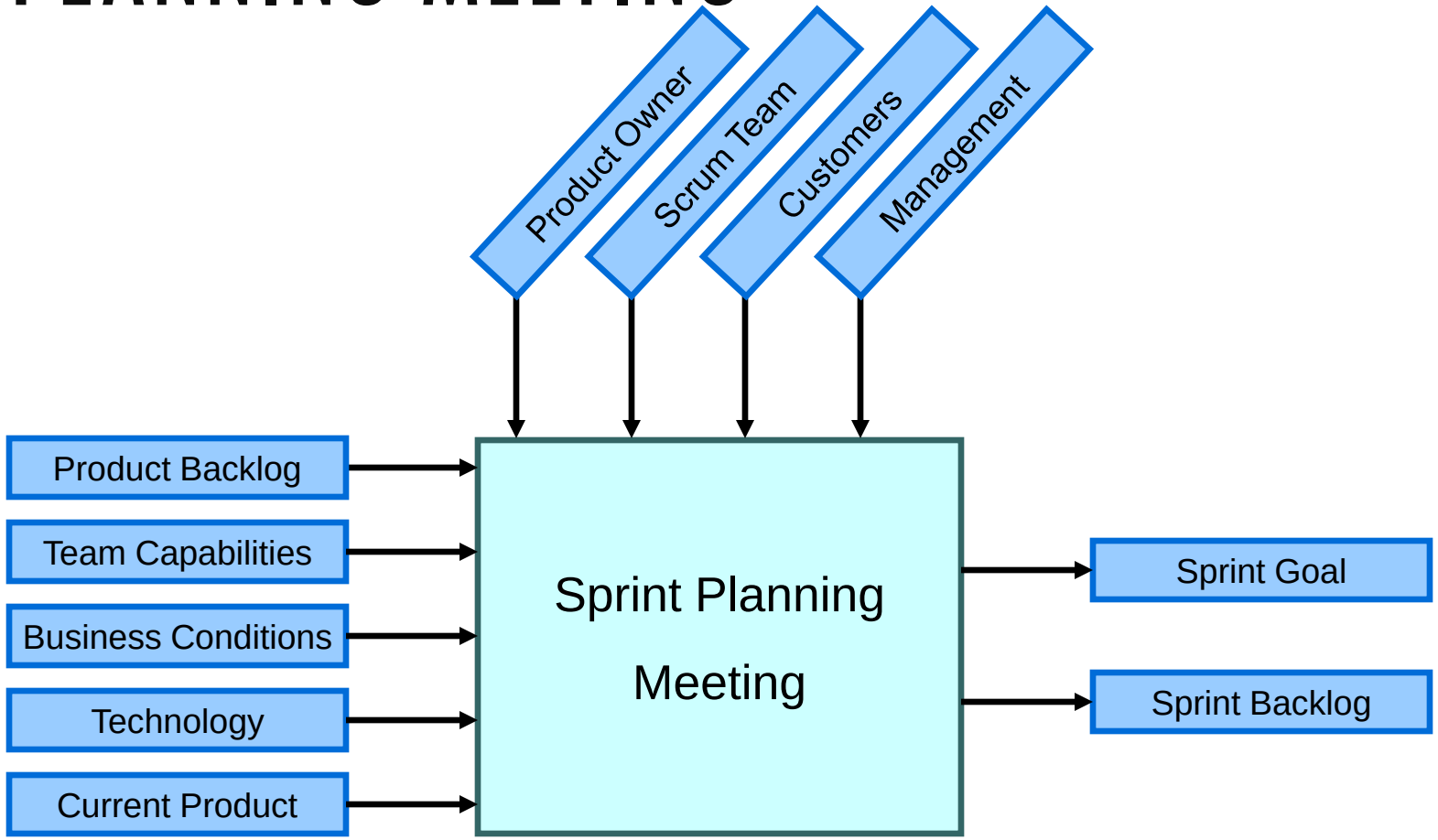
SPRINT

DAILY SCRUM

SPRINT REVIEW MEETING

SPRINT RETROSPECTIVE

SPRING PLANNING MEETING



PARTS OF SPRINT PLANNING MEETING

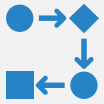
- 1st Part:
  - Creating Product Backlog
  - Determining the Sprint Goal.
  - Participants: Product Owner, Scrum Master, Scrum Team
- 2nd Part:
  - Participants: Scrum Master, Scrum Team
  - Creating Sprint Backlog

PRE-PROJECT/KICKOFF MEETING

A SPECIAL FORM OF SPRINT PLANNING MEETING

MEETING BEFORE THE BEGIN OF THE PROJECT

SPRINT

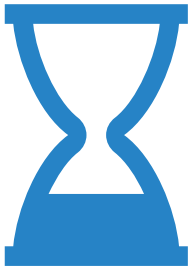


A month-long iteration, during which is incremented a product functionality

NO outside influence can interfere with the Scrum team during the Sprint

Each Sprint begins with the Daily Scrum Meeting

DAILY SCRUM



Parameters

Daily
15-minutes
Stand-up
Not for problem solving

Three questions:

What did you do yesterday
What will you do today?
What obstacles are in your way?

DAILY SCRUM

Is NOT a problem solving session

Is NOT a way to collect information about WHO is behind the schedule

Is a meeting in which team members make commitments to each other and to the Scrum Master

Is a good way for a Scrum Master to track the progress of the Team

SCRUM FAQS

- Why daily?
  - "How does a project get to be a year late?"
    - "One day at a time."
      - Fred Brooks, The Mythical Man-Month.
- Can Scrum meetings be replaced by emailed status reports?
  - No
    - Entire team sees the whole picture every day
    - Create peer pressure to do what you say you'll do

SPRINT REVIEW MEETING

Team presents what it accomplished during the sprint

Typically takes the form of a demo of new features or underlying architecture

Informal

2-hour prep time rule

Participants

Customers
Management
Product Owner
Other engineers

SPRINT RETROSPECTIVE MEETING

Scrum Team only

Feedback meeting

Three questions

- Start
- Stop
- Continue

Don’t skip for the first 5-6 sprints!!!

PRODUCT BACKLOG

A list of all desired work on the project

- Usually a combination of
  - story-based work ("let user search and replace")
  - task-based work ("improve exception handling")

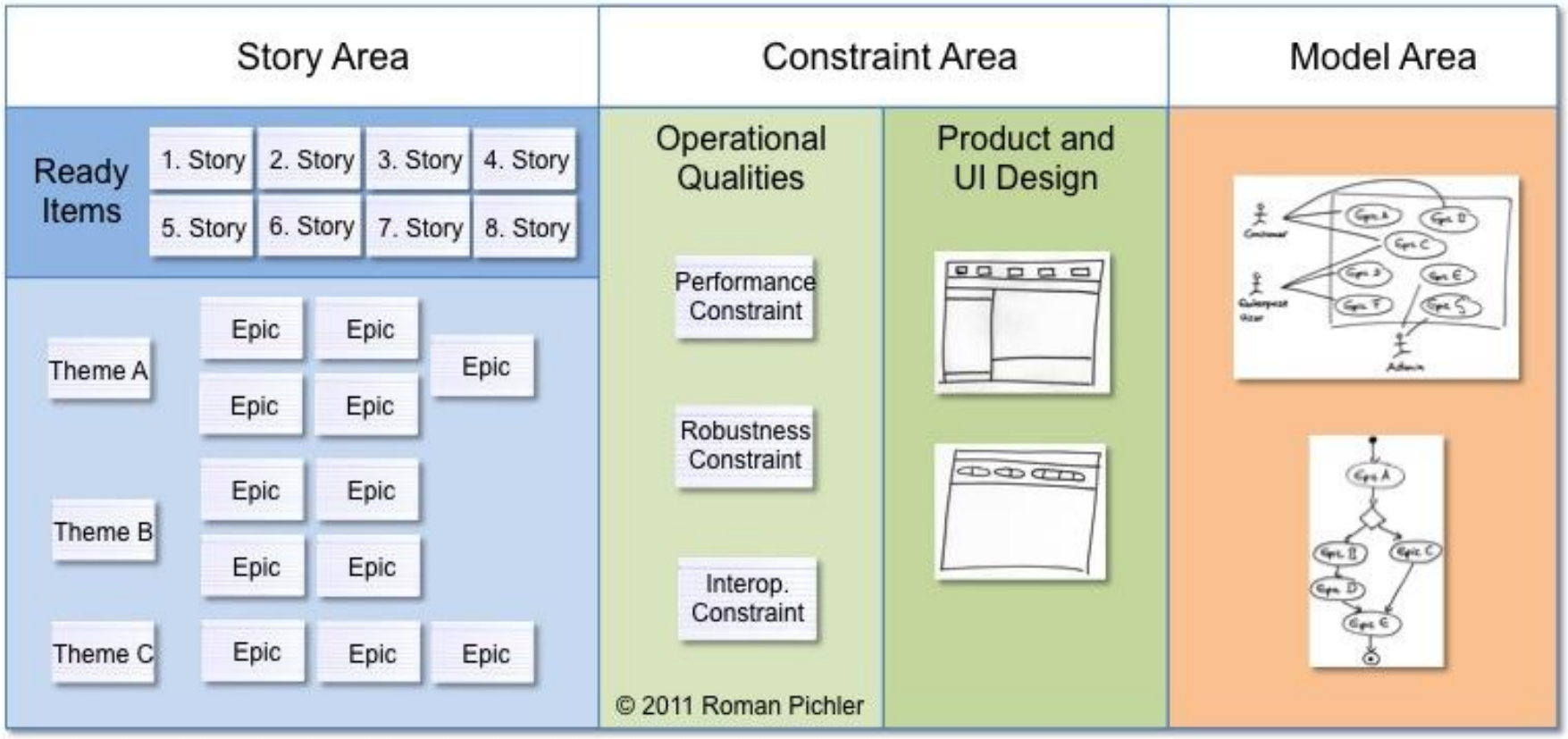
List is prioritized by the Product Owner

- Typically a Product Manager, Marketing, Internal Customer, etc.

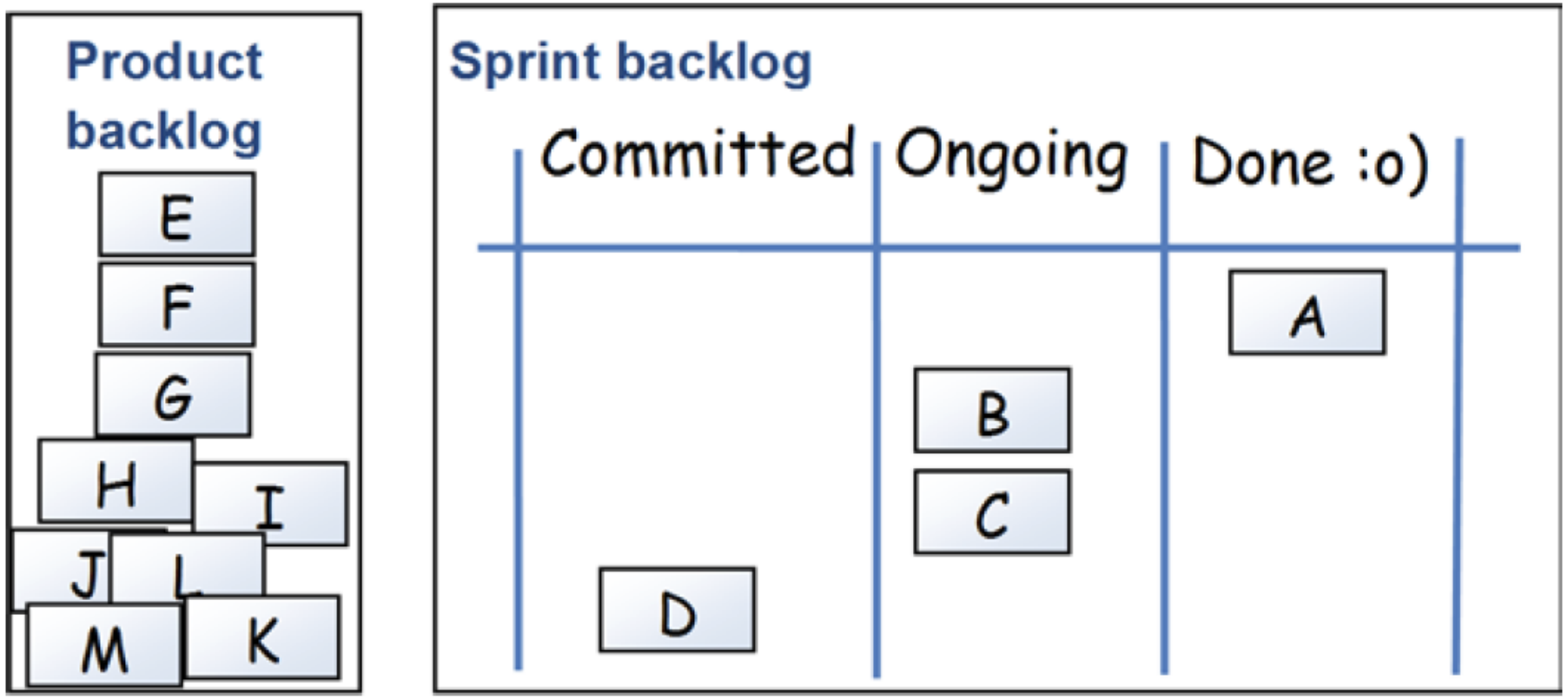
PRODUCT BACKLOG

- Requirements for a system, expressed as a prioritized list of Backlog Items
- Is managed and owned by a Product Owner
- Spreadsheet (typically)
- Usually is created during the Sprint Planning Meeting
- Can be changed and re-prioritized before each PM

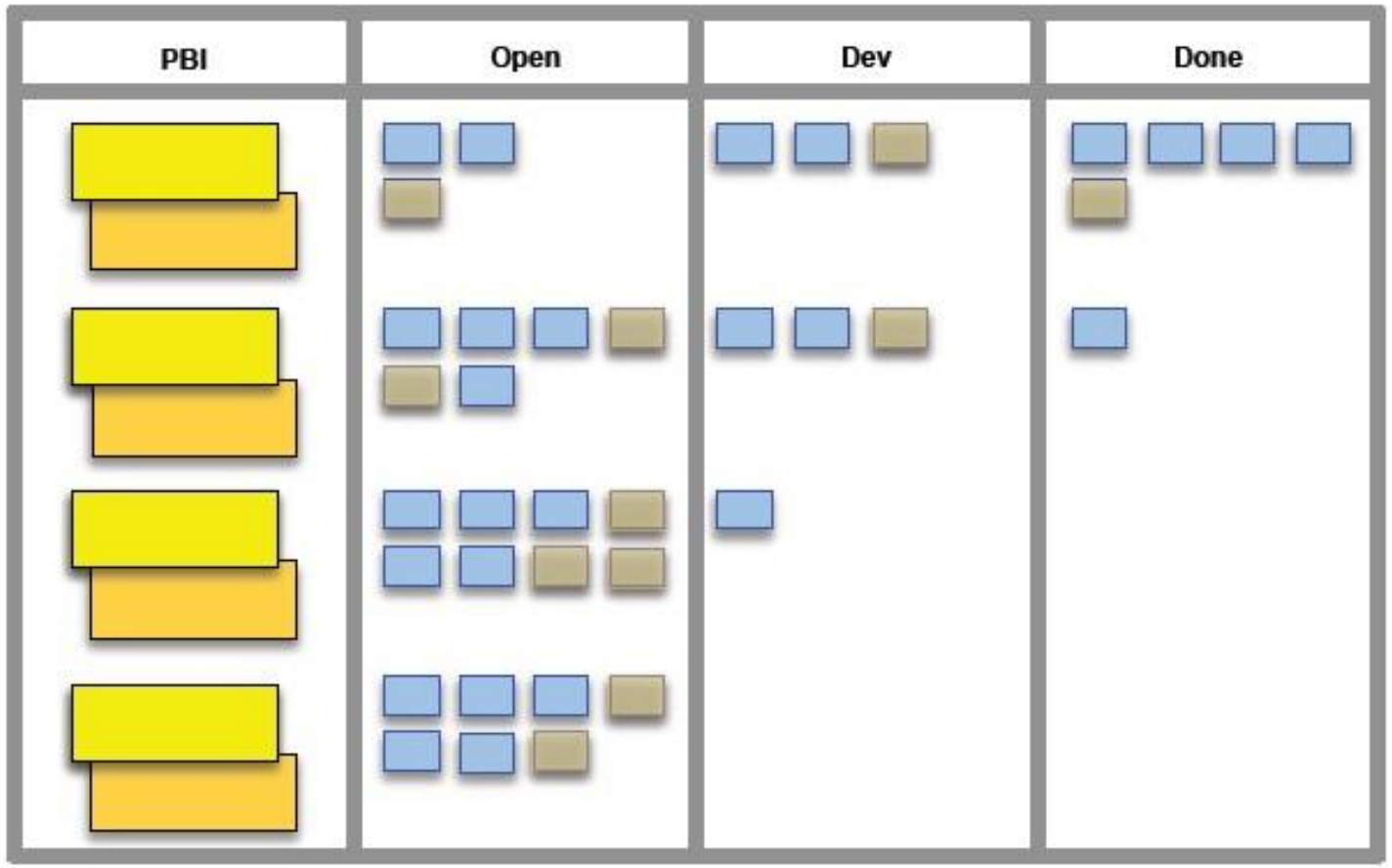
AN EXAMPLE OF PRODUCT BACKLOG



PRODUCT BACKLOG TO SPRINT BACKLOG



KANBAN BOARD



FROM SPRINT GOAL TO SPRINT BACKLOG

Scrum team takes the Sprint Goal and decides what tasks are necessary

Team self-organizes around how they'll meet the Sprint Goal

- Manager doesn't assign tasks to individuals

Managers don't make decisions for the team

Sprint Backlog is created

SPRINT BACKLOG DURING THE SPRINT

Changes

- Team adds new tasks whenever they need to in order to meet the Sprint Goal
- Team can remove unnecessary tasks
- But: Sprint Backlog can only be updated by the team

Estimates are updated whenever there's new information

SPRINT BACKLOG

A subset of Product Backlog Items, which define the work for a Sprint

Is created ONLY by Team members

Each Item has it's own status

Should be updated every day

No more than 300 tasks in the list

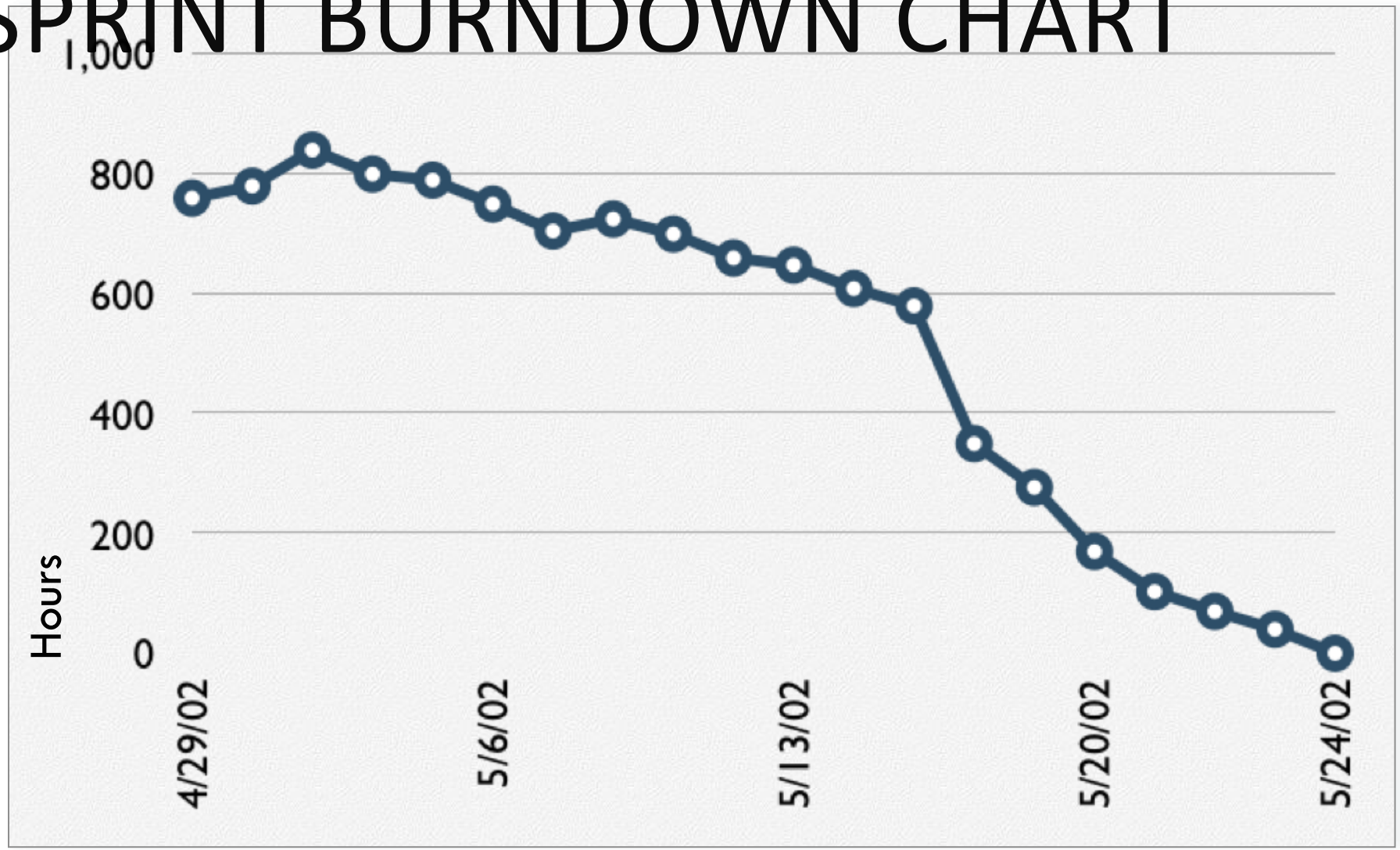
If a task requires more than 16 hours, it should be broken down

Team can add or subtract items from the list. Product Owner is not allowed to do it

A SPRINT BACKLOG

Tasks	Mon	Tues	Wed	Thur	Fri
Code the user interface	8	4	8		
Code the middle tier	16	12	10	4	
Test the middle tier	8	16	16	11	8
Write online help	12				
Write the foo class	8	8	8	8	8
Add error logging			8	4	

A SPRINT BURNDOWN CHART



Tasks	Mon	Tues	Wed	Thur	Fri
Code the user interface	8	4	8		
Code the middle tier	16	12	10	7	
Test the middle tier	8	16	16	11	8
Write online help	12