AOT LAB

**A**gent and **O**bject **T**echnology **Lab**
Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Parma


# Advanced Software Engineering

## Software Development Processes

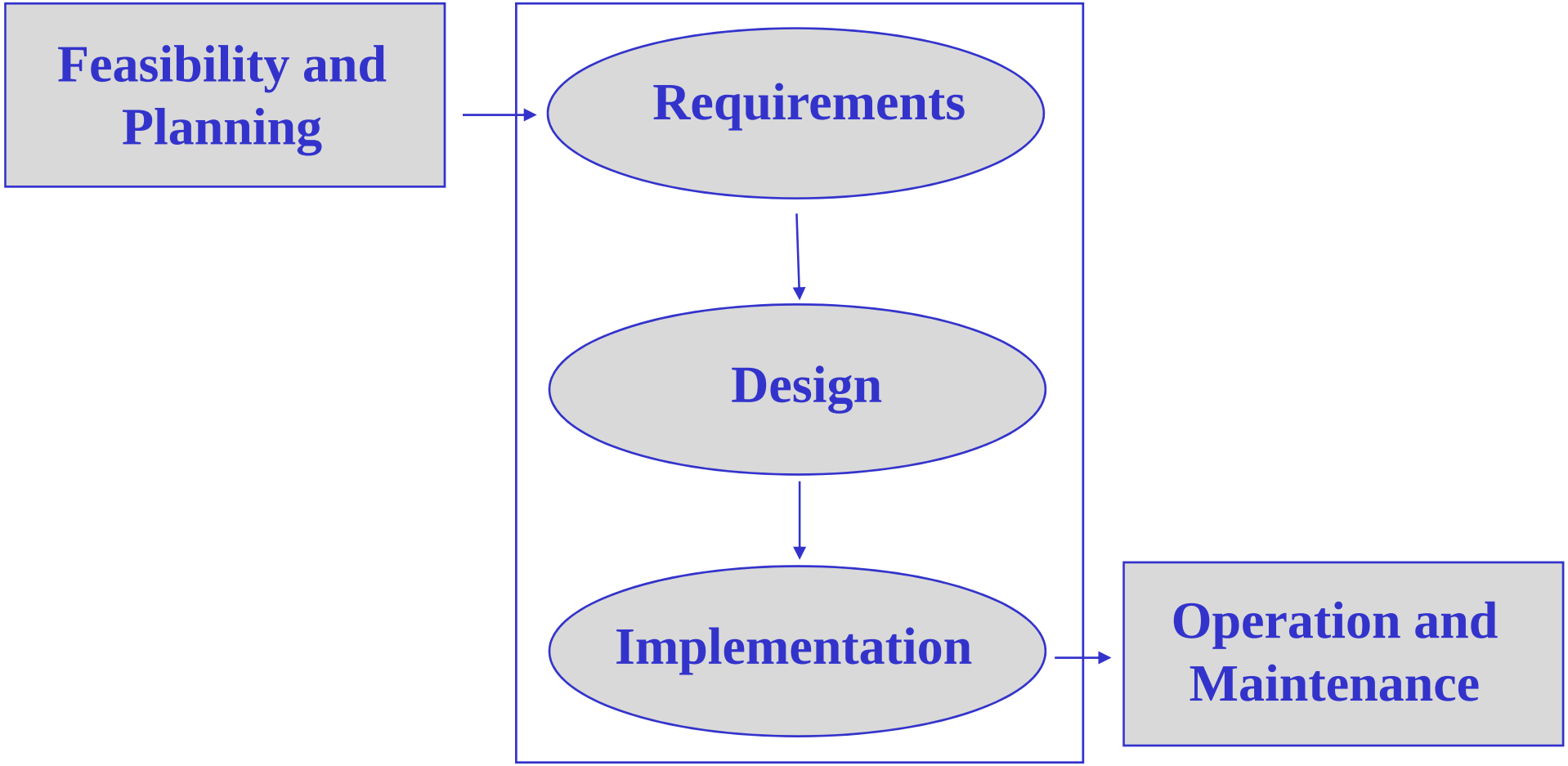
**Prof. Agostino Poggi**

# Software Development Process

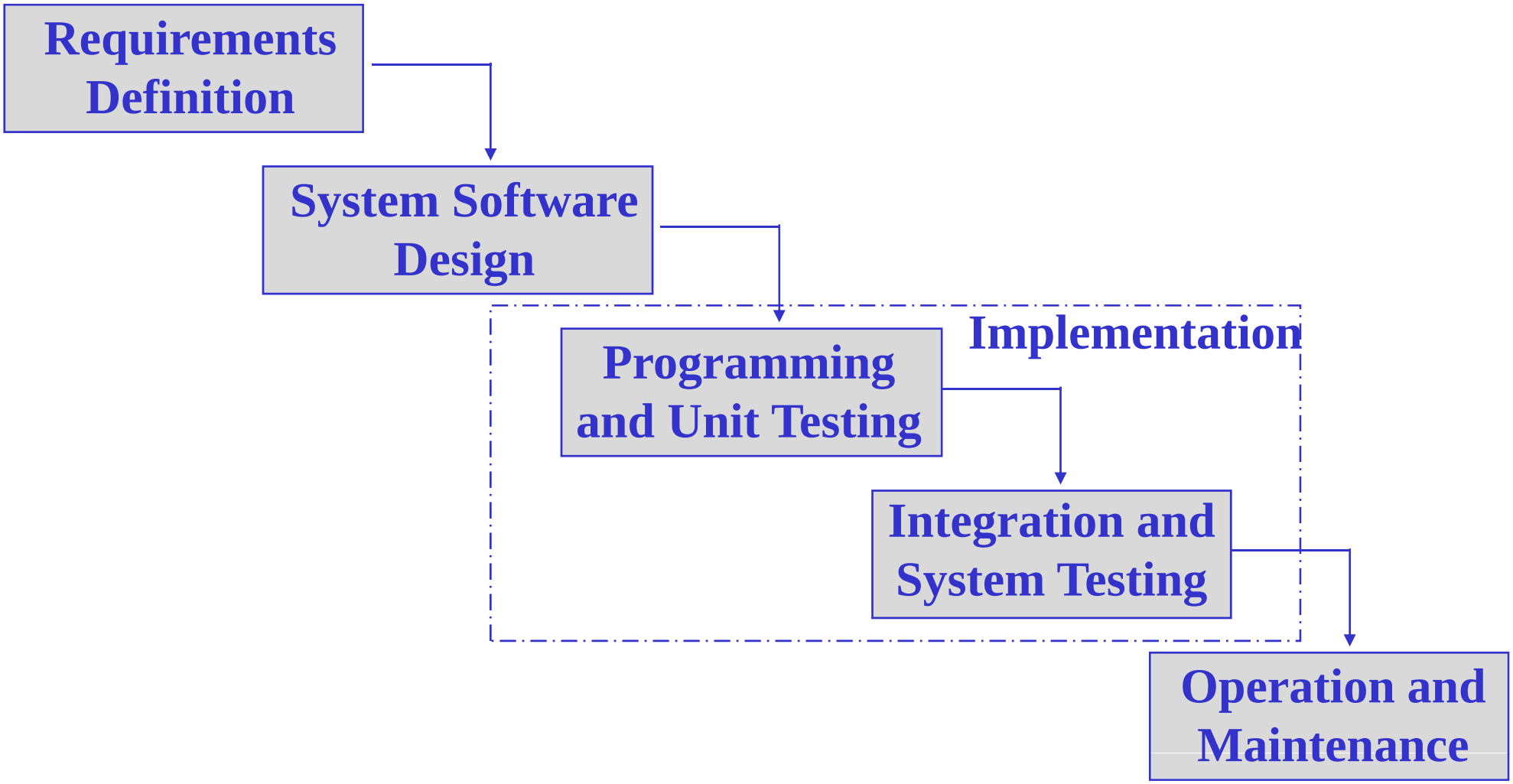
- A process defines **who** is doing **what**, **when** to do it, and **how** to reach a certain goal

- A software development process is a **structured set** of **activities** required to develop a **software system**

# Software Development Process



# Waterfall Model



# Requirements Analysis and Definition

- System services, constraints and goals are established by consultation with system users
- Requirements are then defined in a manner that is understandable by both users and development staff
- This phase can be divided into:
  - Feasibility study (often carried out separately)
  - Requirements analysis
  - Requirements definition
  - Requirements specification

# System and Software Design

- One or more models of the system are defined
  - Models represent the system at different levels of detail
- This phase is divided in two activities:
  - System design
    - Identifies hardware and software components
    - Establishes an overall system architecture
  - Software design
    - Represents the software system functions in a form that can be transformed into one or more executable programs

# Programming and Unit Testing

- System components defined in the design phase are implemented and tested
- This phase is divided in two activities:
  - Software components are realized as a set of programs or program units
    - Written specifically
    - Acquired from elsewhere, or modified
  - Software components are separately tested against specifications

# Integration and System Testing

- Program units are integrated and tested for realizing the complete system

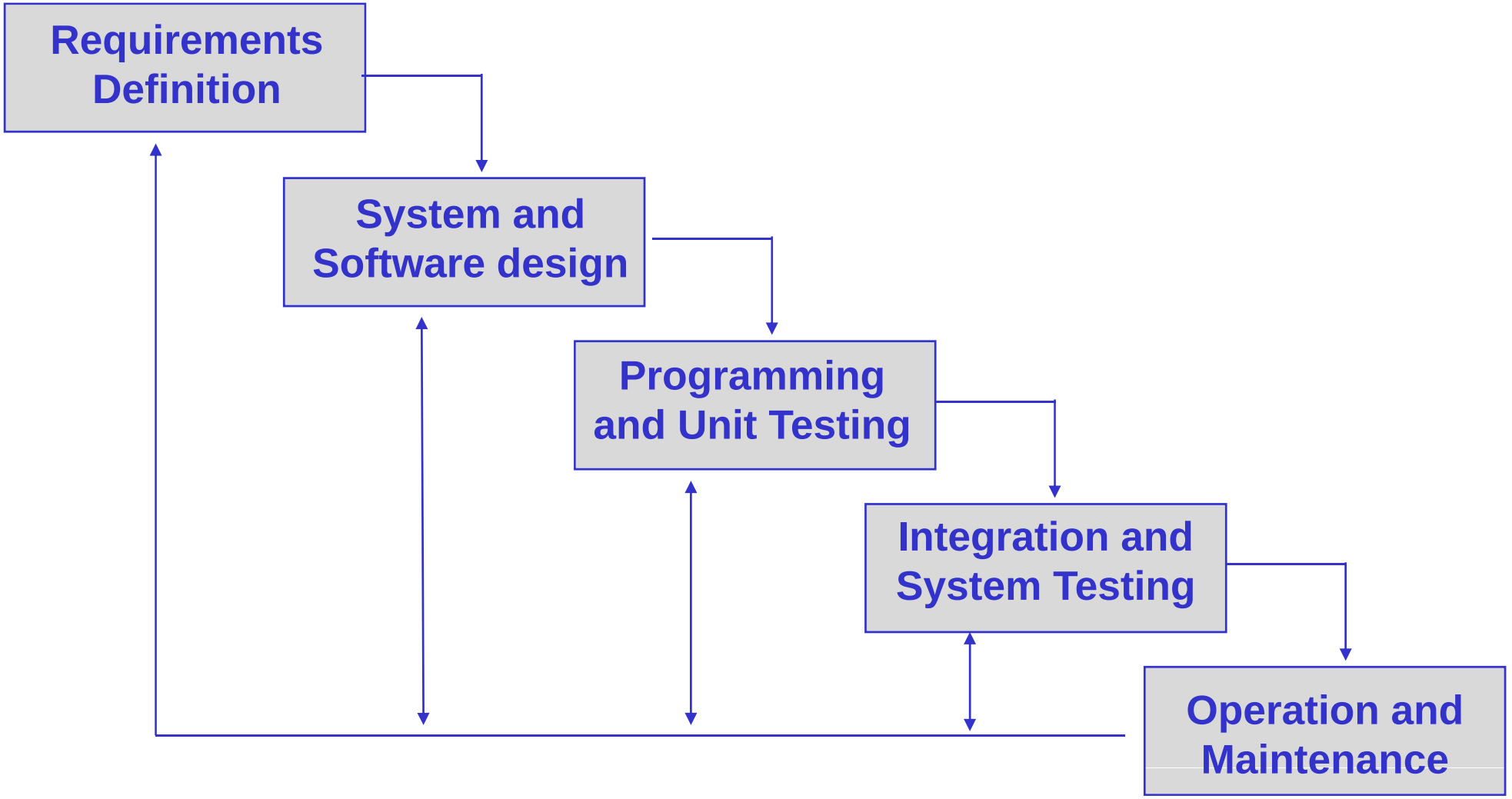
- This phase is divided in four activities:
  - Component integration
  - Integration test
  - System requirements test
  - System deliver to the client

# Operation and Maintenance

- The system is put to use and clients are supported
- This phase is divided in four activities:
  - Operation: the system is put into practical use
  - Maintenance: errors and problems are identified and fixed
  - Evolution: the system evolves over time as requirements change, to add new functions or adapt the technical environment
  - Phase out: the system is withdrawn from service

AOT LAB

# Feedback in the Waterfall Model



# Advantages / Disadvantages

- Advantages
  - Documentation and clearly defined phases
  - Maintenance easier (assuming up-to-date docs available)
- Disadvantages
  - Complete and frozen specification document up-front often not feasible in practice
  - Customer involvement in the first phase only
  - Sequential and complete execution of phases often not desirable
  - Process difficult to control
  - The product becomes available very late in the process
    - Significant risk of building the “wrong” system

# Applicability

- The main drawback of the waterfall model is the difficulty of accommodating change after the process is underway
  - One phase has to be complete before moving onto the next phase
- Only appropriate when the requirements are well-understood and changes will be fairly limited during the design process
- Mostly used for large systems engineering projects where a system is developed at several sites

# Project Failure Causes

- Inaccurate understanding of end-user needs
- Inability to deal with changing requirements
- Modules that do not fit together
- Software that is hard to maintain or extend
- Late discovery of serious project flaws
- Poor software quality
- Unacceptable software performance
- Untrustworthy build-and-release processes

# Project Failure Causes

- Ad hoc requirements management
- Ambiguous and imprecise communication
- Brittle architectures
- Overwhelming complexity
- Undetected inconsistencies in requirements, designs and implementations
- Insufficient testing
- Subjective assessment of project status

# Agile Software Development

- Agile software development is a conceptual framework for software engineering that promotes development iterations throughout the life-cycle of the project
- Agile methods emphasize face-to-face communication over written documents
  - Most agile teams are located in a single open office sometimes referred to as a bullpen
  - At a minimum, a team includes programmers and their "customers"
  - Agile methods also emphasize working software as the primary measure of progress producing very little written documentation relative to other methods

# Agile Software Development

- Software developed is based on unit called iterations, which usually may last from one to four weeks
- Each iteration is an entire software project: including planning, requirements analysis, design, coding, testing, and documentation
- An iteration may not add enough functionality to warrant releasing the product to market
- Iteration goal is to have an available release (without bugs) at the end of each iteration
- At the end of each iteration, the team re-evaluates project priorities

# Agile Methodologies Principles

- Customer involvement
- Incremental delivery
- A focus on people, not the process
- Embracing of change and maintaining simplicity
- Best suited for small or medium sized systems

# Agile Manifesto

- Customer satisfaction by rapid, continuous delivery of useful software
- Working software is delivered frequently (weeks rather than months)
- Working software is the principal measure of progress
- Even late changes in requirements are welcomed
- Close, daily, cooperation between business people and developers

# Agile Manifesto

- Face-to-face conversation is the best form of communication
- Projects are built around motivated individuals, who should be trusted
- Continuous attention to technical excellence and good design
- Simplicity
- Self-organizing teams
- Regular adaptation to changing circumstances

# Agile Methodologies Best Practices

- Developing software iteratively
- Managing requirements
- Use of component-based architectures
- Visually modeling software
- Continuously verifying software quality
- Controlling changes to software

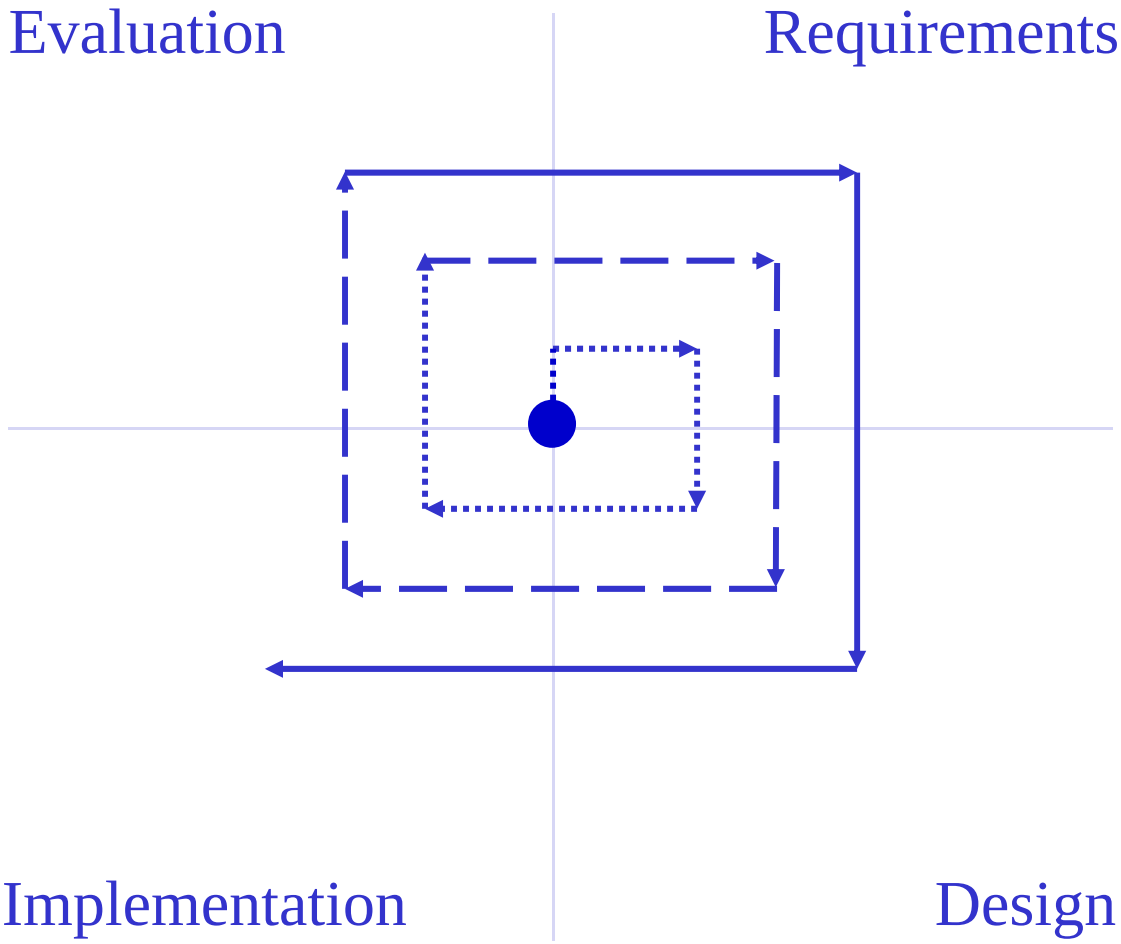
# Iterative Refinement

- Iterative development is a rework scheduling strategy in which time is set aside to revise and improve parts of the system
- The goal of iterative refinement (evolutionary development) is to get something working as quickly as possible
- The initial implementation is refined until system is complete on the basis of client and user comment

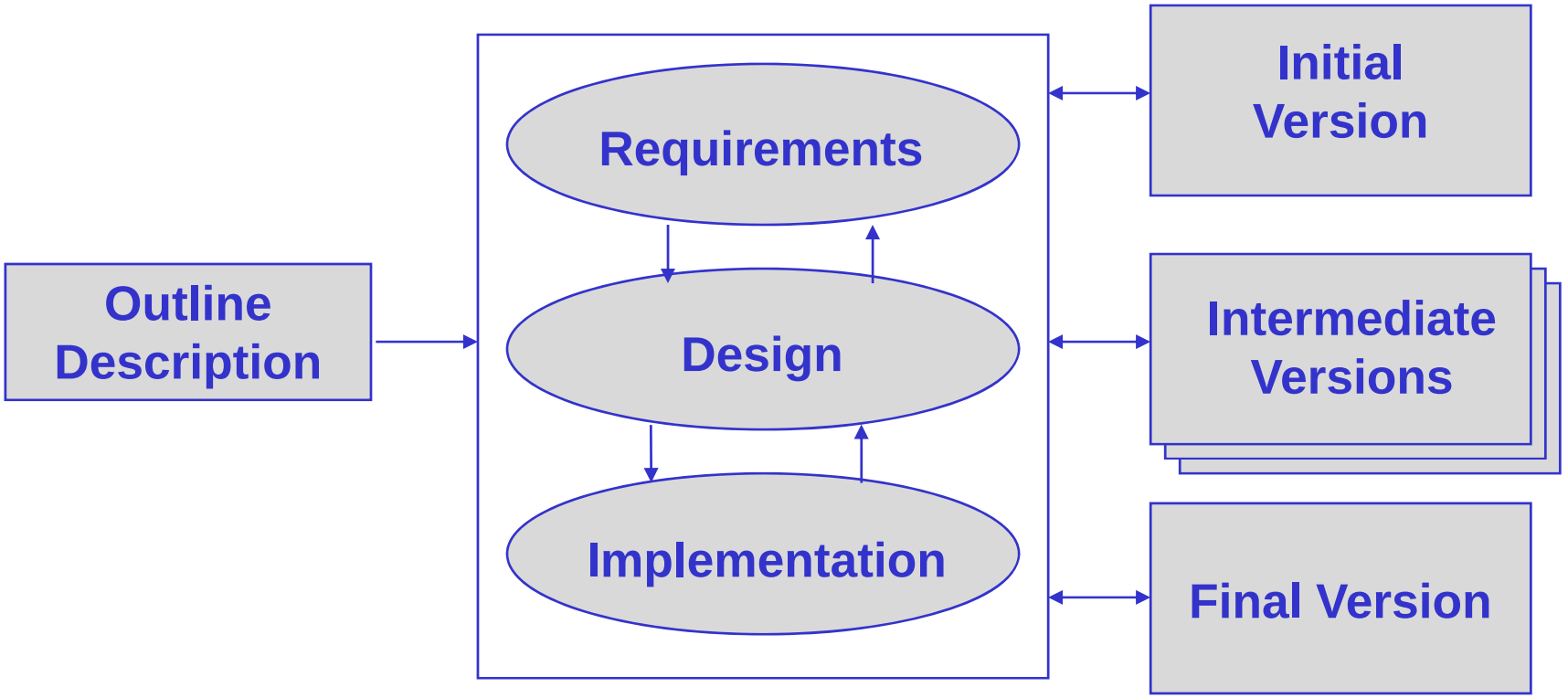
# Iterative Refinement Techniques

- Vaporware:  user interface mock-up
- Throw-away software components
- Dummy modules
- Rapid prototyping: RAD tools
- Incremental refinement

# Iterative Refinement



# Iterative Refinement



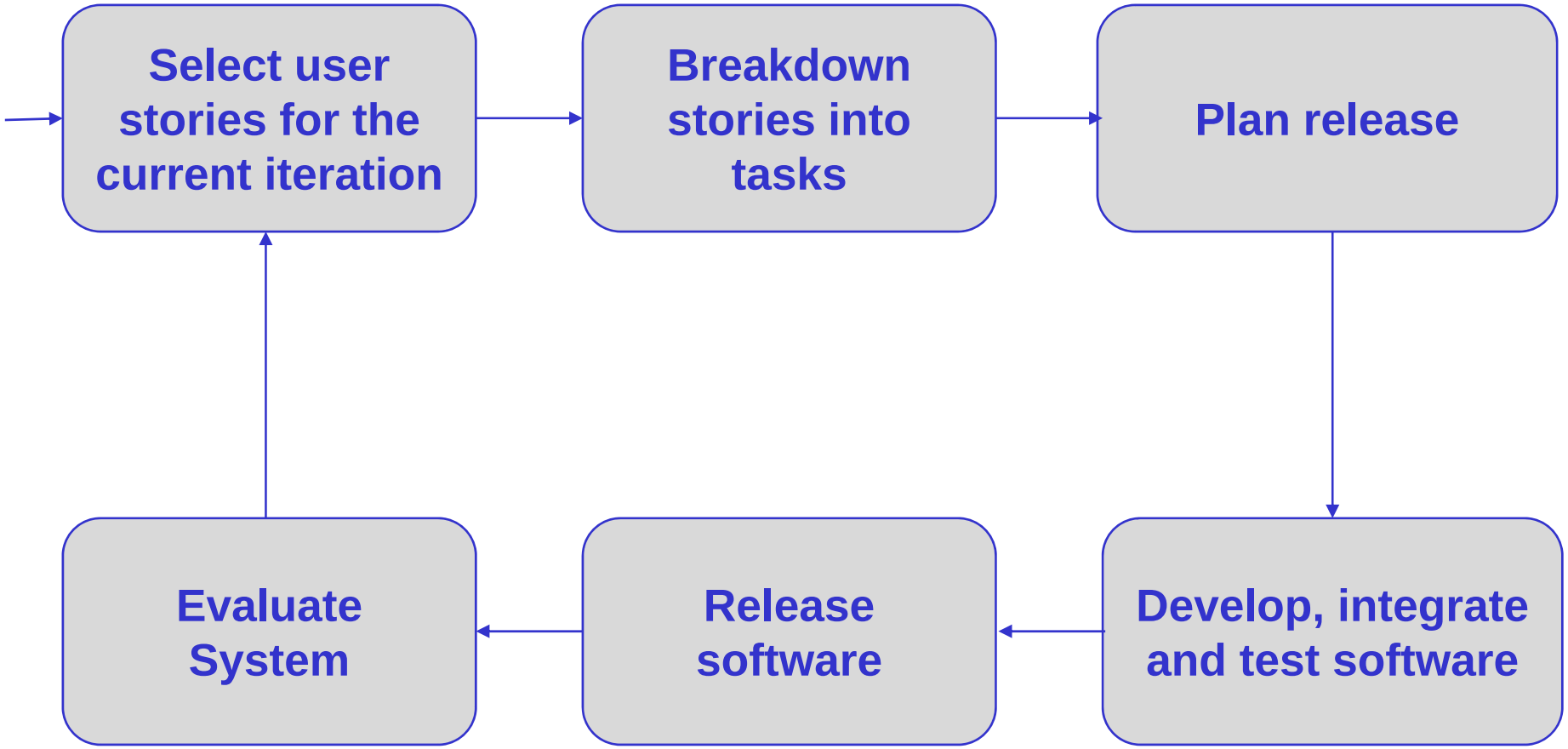
# Applicability

- The main problems of iterative refinement are:
  - Lack of process visibility
  - Systems are often poorly structured
  - Special skills (e.g. in languages for rapid prototyping) may be required
- Iterative refinement can be used with success for:
  - Small or medium-size interactive systems
  - Parts of large systems (e.g., the user interface)
  - Short-lifetime systems

# Extreme Programming

- Extreme Programming (XP) is probably the best known and most widely used agile method
- In XP all scenarios are represented by user stories implemented as a series of tasks
- Programmers work in pairs and develop tests for each task before writing code and all tests must be executed when new code is integrated into the system
- XP is based on four activities: listening, design, coding and testing

# Extreme Programming

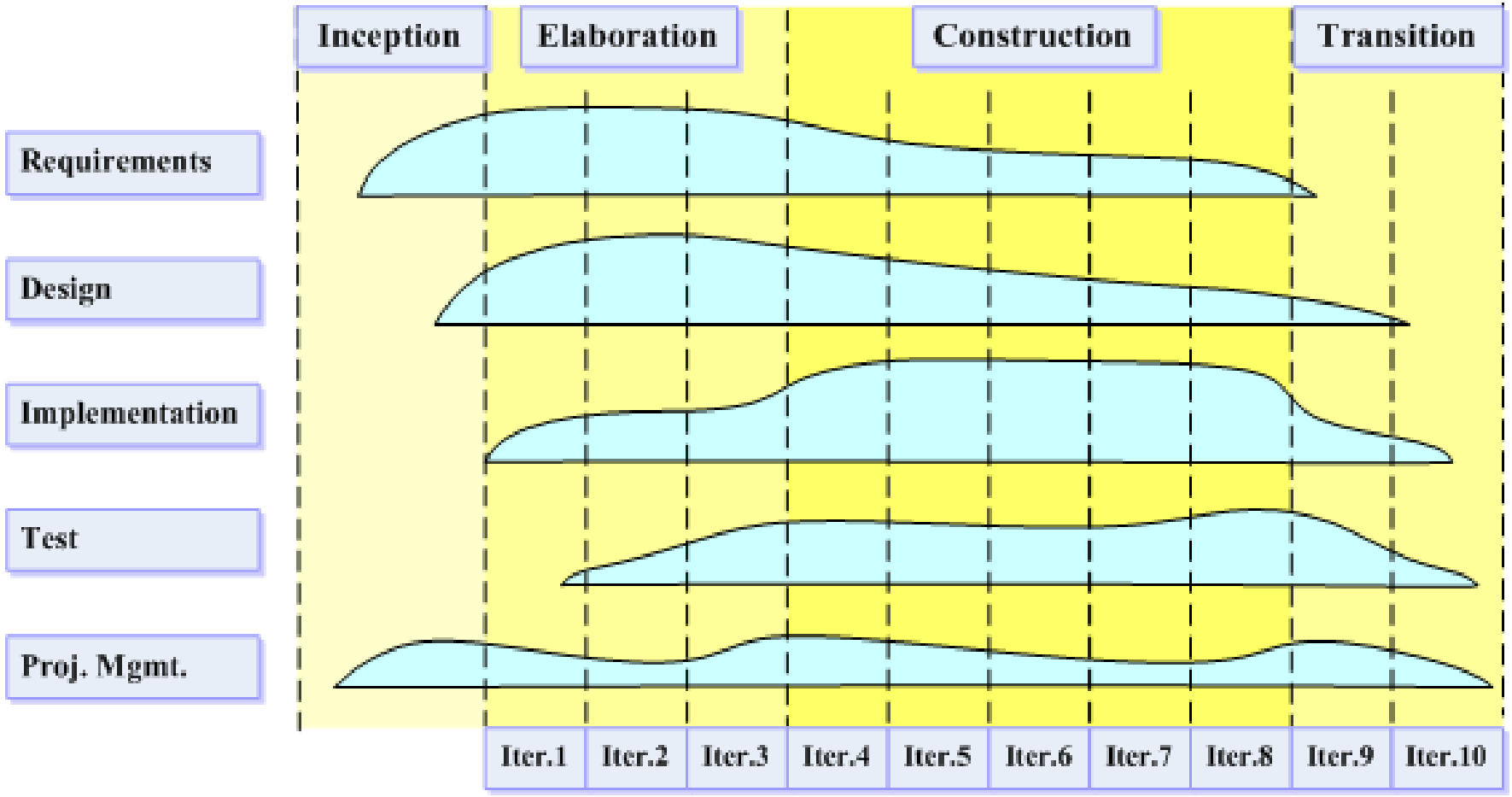


# Unified Process

- Unified Process (UP) is not simply a process, but rather an extensible framework which should be customized for specific organizations or projects
- UP is guided by use cases and is based on early risks identification and management
- UP is architecture-centric because the architecture sits at the heart of the project efforts to shape the system
- UP is an development process where each iteration results in an increment, which is a system release that:
  - Contains added or improved functionality
  - Includes work in most of the process disciplines

AOT LAB

# Unified Process

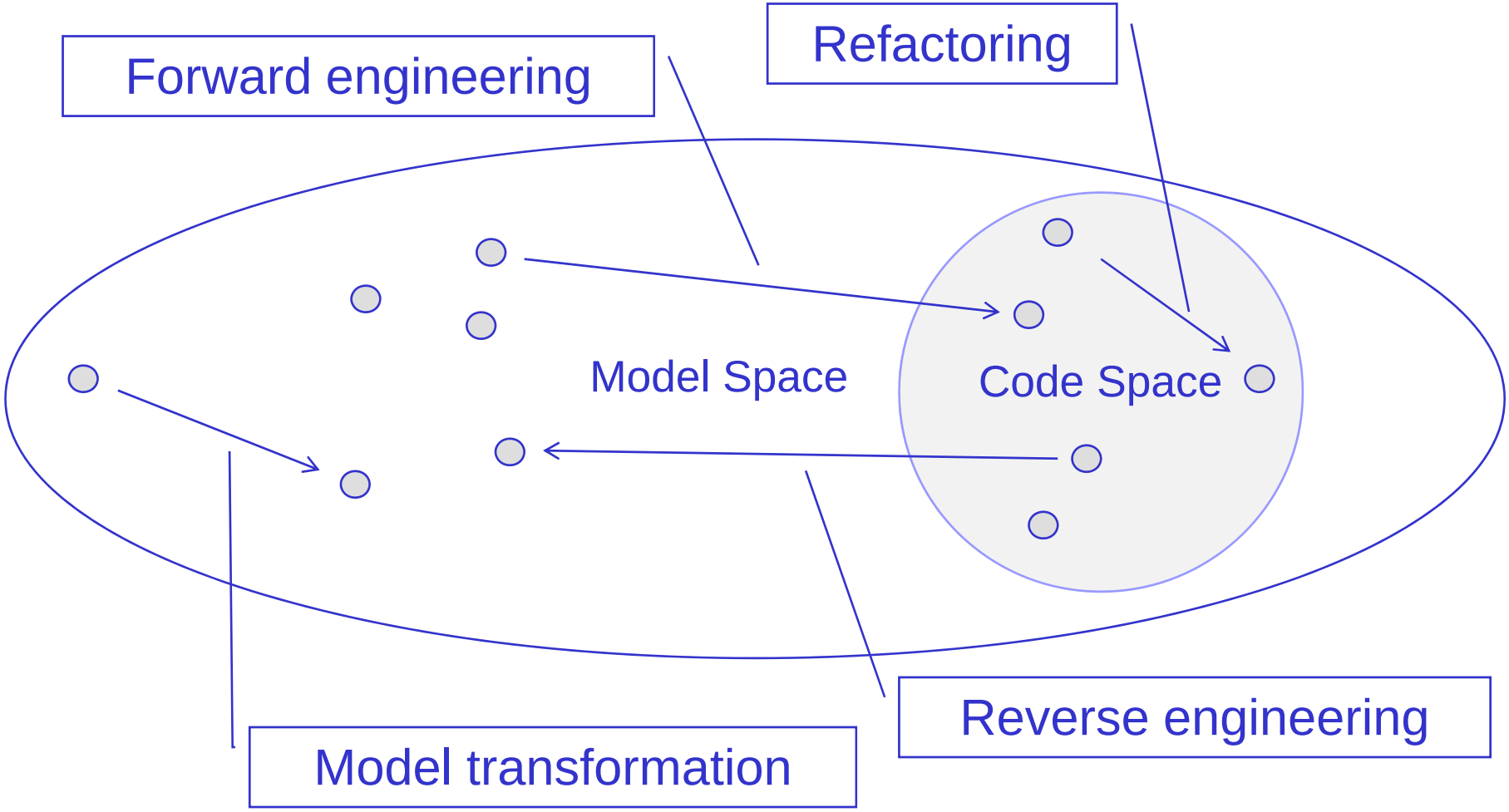


# Model Driven Development

- Model driven development refers to the systematic use of models as primary engineering artifacts throughout the development lifecycle
- Model driven development is based on the transformation of a model to an executable program
  - Sometimes models are constructed to a certain level of detail, and then code is written by hand in a separate step
  - Sometimes complete models are built including executable actions
    - Code can be generated from the models, ranging from system skeletons to complete, deployable products

AOT LAB

# Model Transformation



# Applicability

- The main drawback of the model driven development are:
  - Need for specialized skills and training to apply the technique
  - Difficult to formally specify some aspects of the system such as the user interface
- Suitable for critical systems especially those where a safety or security case must be made before the system is put into operation

# Model Driven Architecture

- Model Driven Architecture (MDA) is an approach to software development that provides a set of guidelines for structuring specifications expressed as models for the realization of system for different technological platforms
- MDA provides a means for using models to direct the course of the phases of the development of a system
- In particular, MDA uses three types of model:
  - Platform Independent Model (PIM)
  - Computational Independent Model (CIM)
  - Platform Specific Model (PSM)

# Model Driven Architecture