Chapter 1: Introduction to Systems Analysis and Design

WILEY

Learning Objectives

- Systems development life cycle
  - Identify the four phases
  - How it came about
  - Methodology alternatives
- Team roles & skill sets
- Object-oriented systems characteristics
- Object-oriented systems analysis & design
- The Unified Process & its extensions
- The Unified Modeling Language (UML)

Introduction

- Why do we need a formal process?
  - Failures occur (too) often
  - Creating systems is not intuitive
  - Projects are late, over budget or delivered with fewer features than planned
- The System Analyst is the key person
  - Designs a system to add value
  - Must understand the business processes
  - Job is rewarding, yet challenging
  - Requires specific skill sets

Systems Development Life Cycle (SDLC)

The SDLC Process

- The process consists of four phases
- Each phase consists of a series of steps
- Each phase is documented (deliverables)
- Phases are executed sequentially, incrementally, iteratively or in some other pattern

Questions to be Answered

- Planning phase
  - Why should we build this system?
  - What value does it provide?
  - How long will it take to build?
- Analysis phase
  - Who will use it?
  - What should the system do for us?
  - Where & when will it be used?
- Design phase
  - How should we build it?

SDLC: The Planning Phase

1. Project Initiation
   - Develop/receive a system request describing the need and business value
   - Conduct a feasibility analysis
2. Project Management
   - Develop the work plan
   - Staff the project
   - Monitor & control the project

SDLC: The Analysis Phase

1. Develop an analysis strategy
   - Model the current (as-is) system
   - Formulate the new (to-be) system
   - Collect shortcomings of the as-is, and what to do differently in the to-be.

SDLC: The Analysis Phase

1. Develop an analysis strategy
2. Gather the requirements
   - Develop a system concept
   - Create a business model to represent:
     - Business data
     - Business processes
3. Develop a system proposal

SDLC: The Design Phase

1. Develop a design strategy
2. Design architecture and interfaces
3. Develop databases and file specifications
4. Develop the program design to specify:
   - What programs to write
   - What each program will do

SDLC: The Implementation Phase

1. Construct the system
   - Build it (write the programming code)
   - Test it
2. Install system
   - Train the users
3. Support the system (maintenance)

SDLC: Methodologies

- Methodology: a formalized approach to implementing the SDLC
- Categories
  - Process oriented
  - Data centered
  - Object-oriented
  - Structured
  - Rapid action development
  - Agile development

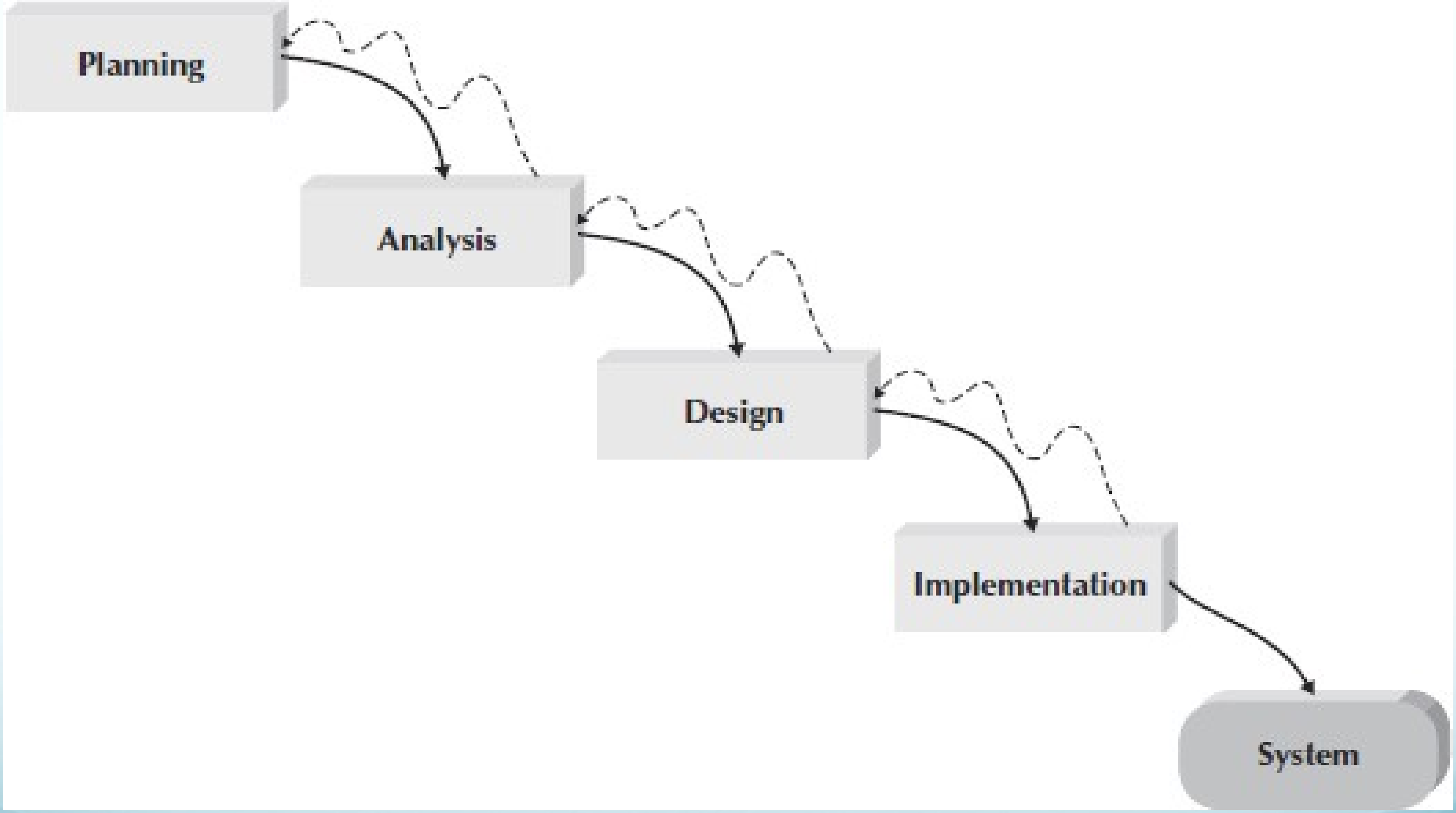
Classes of Methodologies

- Structured Development
  - Waterfall Development
  - Parallel Development
- Rapid Application Development
  - Phased
  - Prototyping
- Agile Development
  - eXtreme Programming
  - SCRUM

Waterfall

- Execute the four steps in order, planning, analysis, design, implementation
- Finish each before starting the next...
- Mostly, but recognizes that you must sometimes back up and revise an earlier step

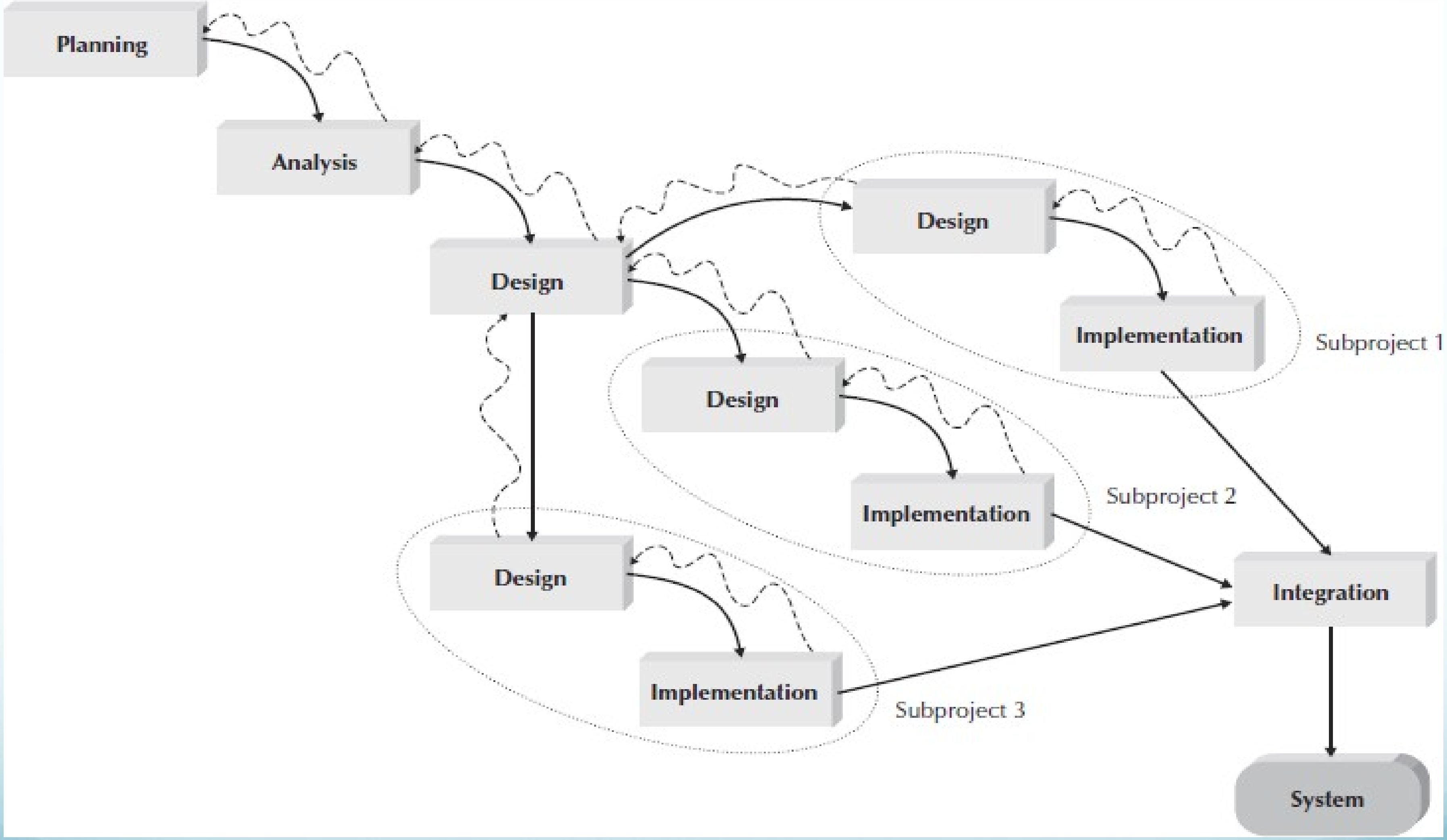
Waterfall



Parallel

- Like waterfall, but split at design step into subprojects
- Pursue each subproject
- Integrate the subprojects into the whole

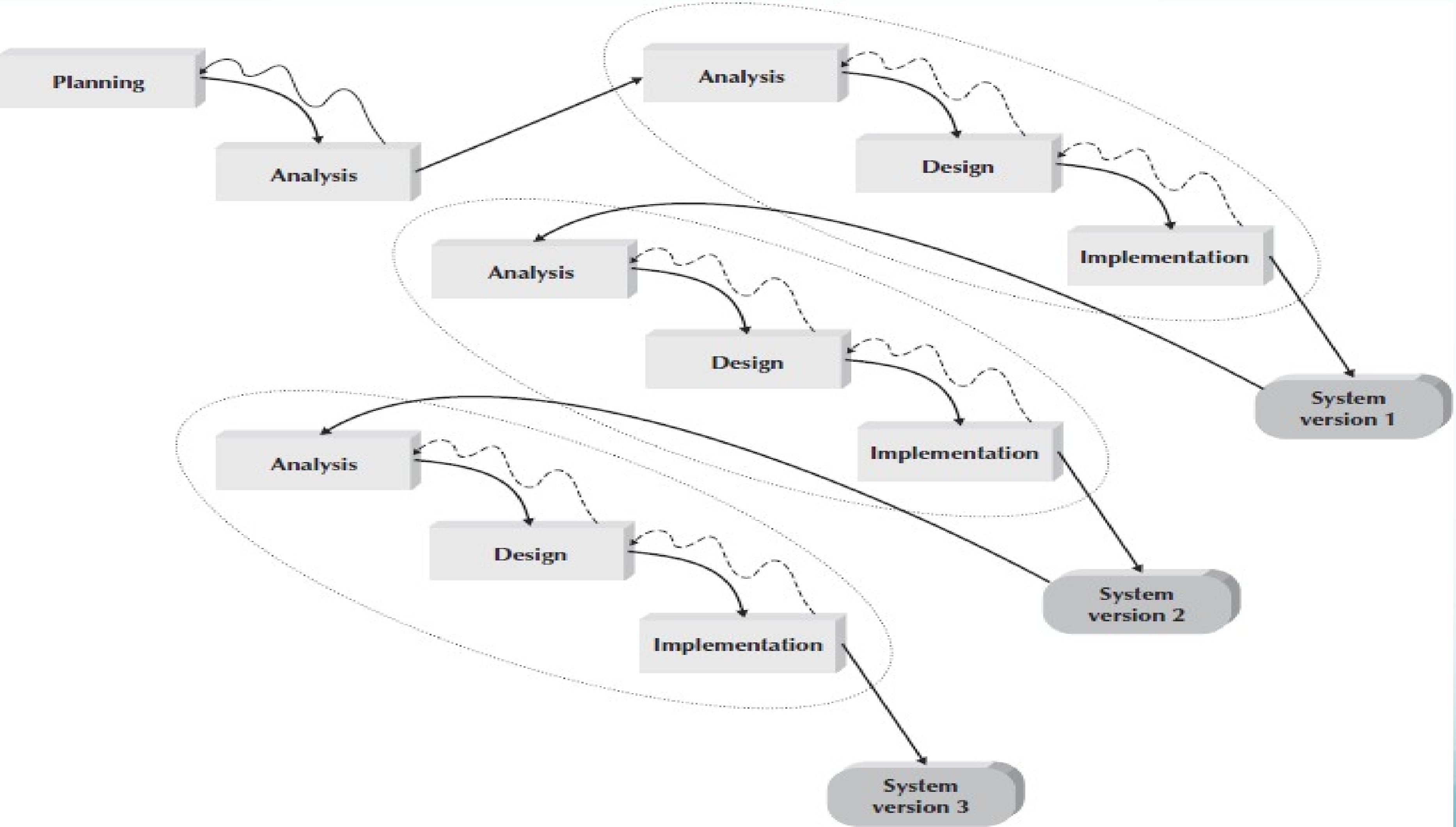
Parallel


Planning
Analysis
Design
Design
Implementation
Subproject 1
Design
Implementation
Subproject 2
Design
Implementation
Subproject 3
Integration
System

Phased and Prototyping

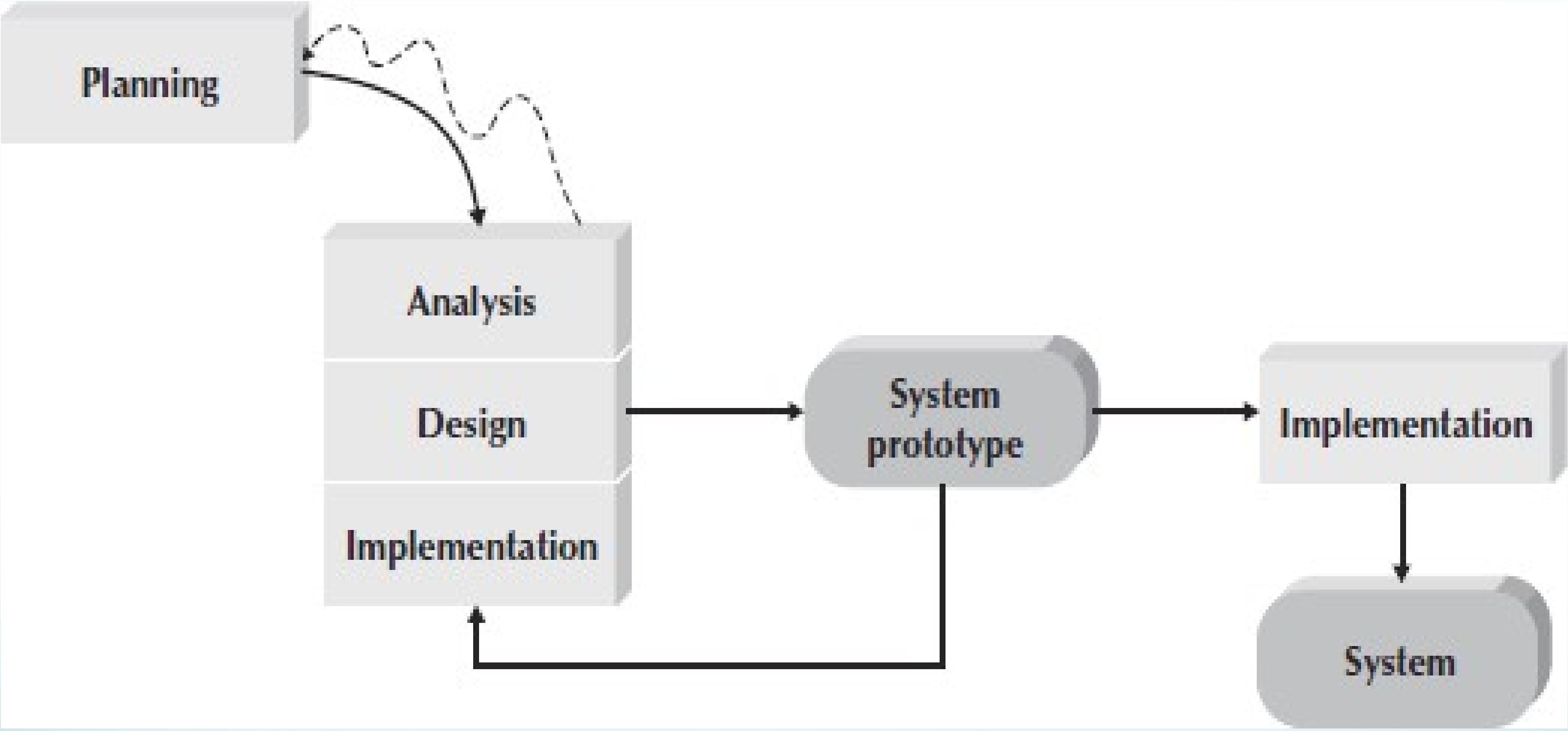
- Phased uses a series of prototypes
- Users give feedback on successive versions
- Prototyping is similar, but versions are less complete
- Throw-away prototypes are shells. Show how the system would look, but do not function

Phased



PowerPoint Presentation for Dennis, Wixom, & Tegarden *Systems Analysis and Design with UML, 5th Edition*

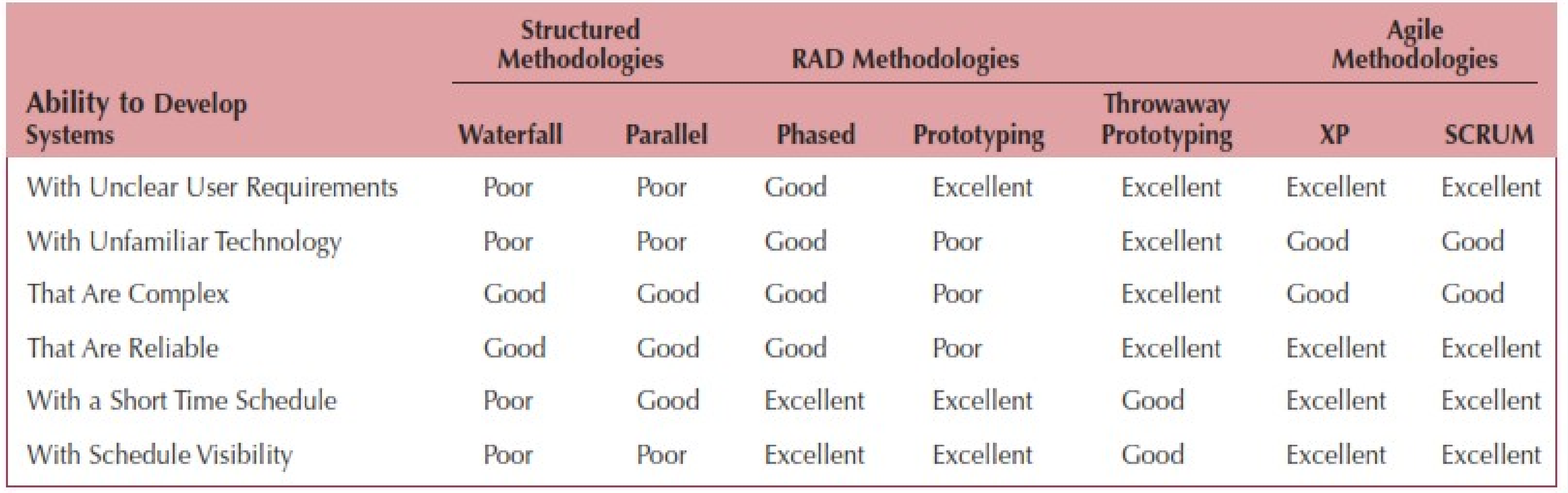
Prototyping



Agile and Extreme

- Attempt to minimize management
- Programmers work in teams, which meet periodically to coordinate
- Steps performed repeatedly
- Progress means more working code

Which Methodology to Use?



Ability to Develop Systems	Structured Methodologies		RAD Methodologies			Agile Methodologies	
	Waterfall	Parallel	Phased	Prototyping	Throwaway Prototyping	XP	SCRUM
With Unclear User Requirements	Poor	Poor	Good	Excellent	Excellent	Excellent	Excellent
With Unfamiliar Technology	Poor	Poor	Good	Poor	Excellent	Good	Good
That Are Complex	Good	Good	Good	Poor	Excellent	Good	Good
That Are Reliable	Good	Good	Good	Poor	Excellent	Excellent	Excellent
With a Short Time Schedule	Poor	Good	Excellent	Excellent	Good	Excellent	Excellent
With Schedule Visibility	Poor	Poor	Excellent	Excellent	Good	Excellent	Excellent

The Systems Analyst: Skills

- Agents of change
  - Identify ways to improve the organization
  - Motivate & train others
- Skills needed:
  - Technical: must understand the technology
  - Business: must know the business processes
  - Analytical: must be able to solve problems
  - Communications: technical & non-technical audiences
  - Interpersonal: leadership & management
  - Ethics: deal fairly and protect confidential information

The Systems Analyst: Roles

- Business Analyst
  - Focuses on the business issues
- Systems Analyst
  - Focuses on the IS issues
- Infrastructure Analyst
  - Focuses on the technical issues
- Change Management Analyst
  - Focuses on the people and management issues
- Project Manager
  - Ensures that the project is completed on time and within budget

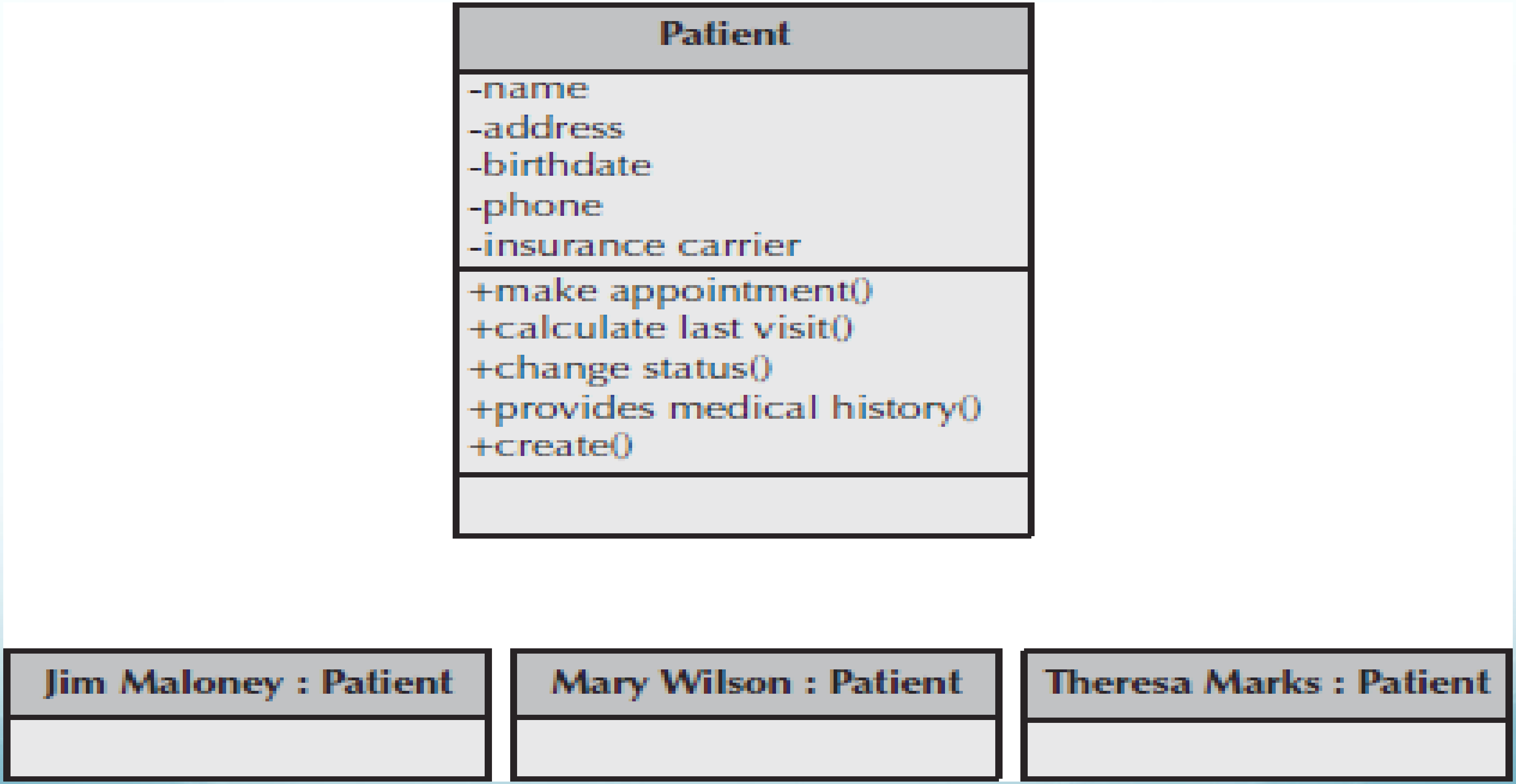
Object-Oriented Systems Analysis & Design

- Attempts to balance data and process
- Utilizes the Unified Modeling Language (UML) and the Unified Process
- Characteristics of OOAD:
  - Use-case Driven
  - Architecture Centric
  - Iterative and Incremental

Characteristics of Object-Oriented Systems

- Classes & Objects
  - Object (instance): instantiation of a class
  - Attributes: information that describes the class
  - State: describes its values and relationships at a point in time

Characteristics of Object-Oriented Systems



Patient

-name
-address
-birthdate
-phone
-insurance carrier

+make appointment()
+calculate last visit()
+change status()
+provides medical history()
+create()

Jim Maloney : Patient

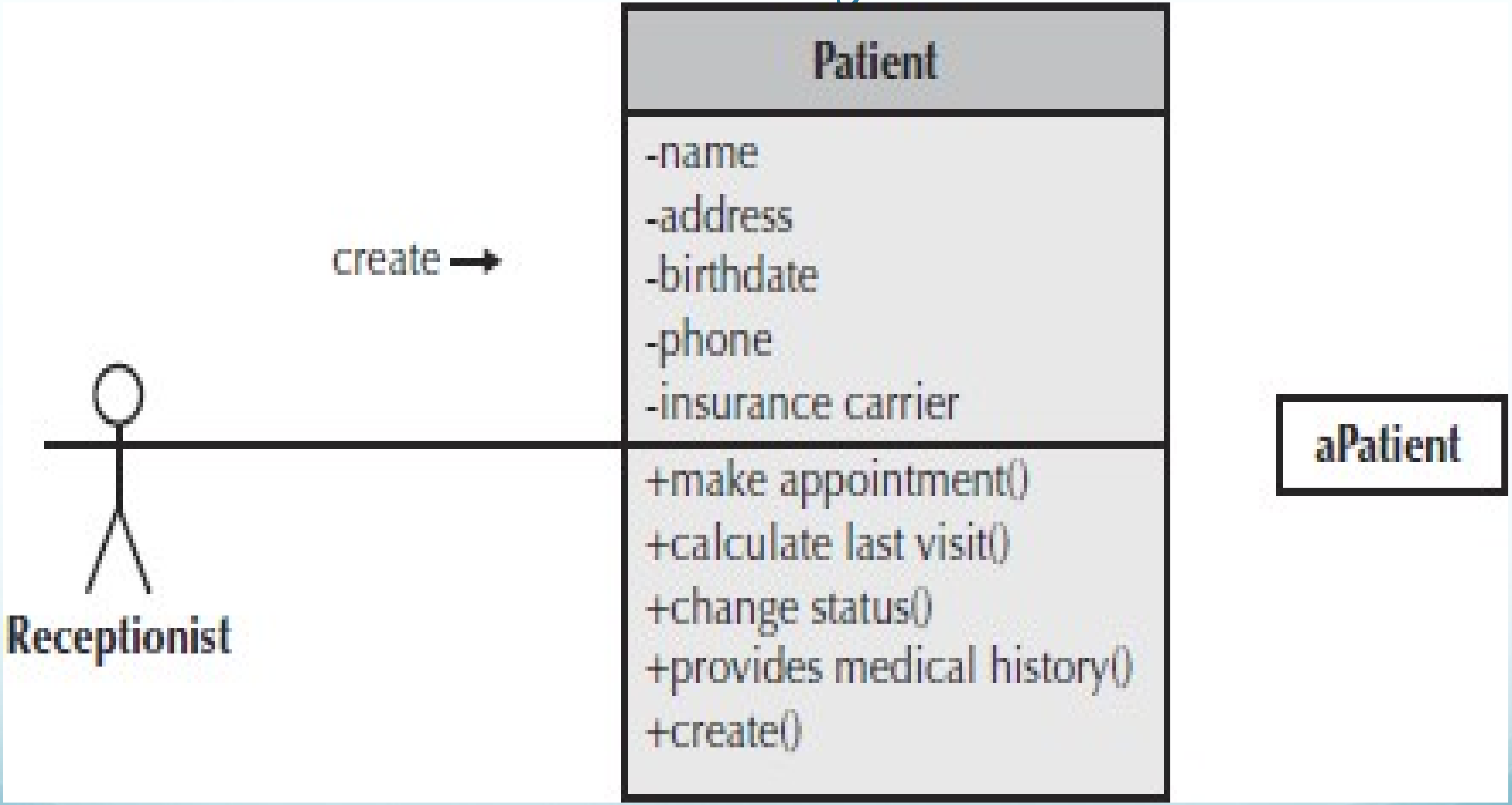
Mary Wilson : Patient

Theresa Marks : Patient

Characteristics of Object-Oriented Systems

- Methods & Messages
  - Methods: the behavior of a class
  - Messages: information sent to an object to trigger a method (procedure call)
  - It’s a “message” because the parameter values carry information to the object when the method is called.

Characteristics of Object-Oriented Systems



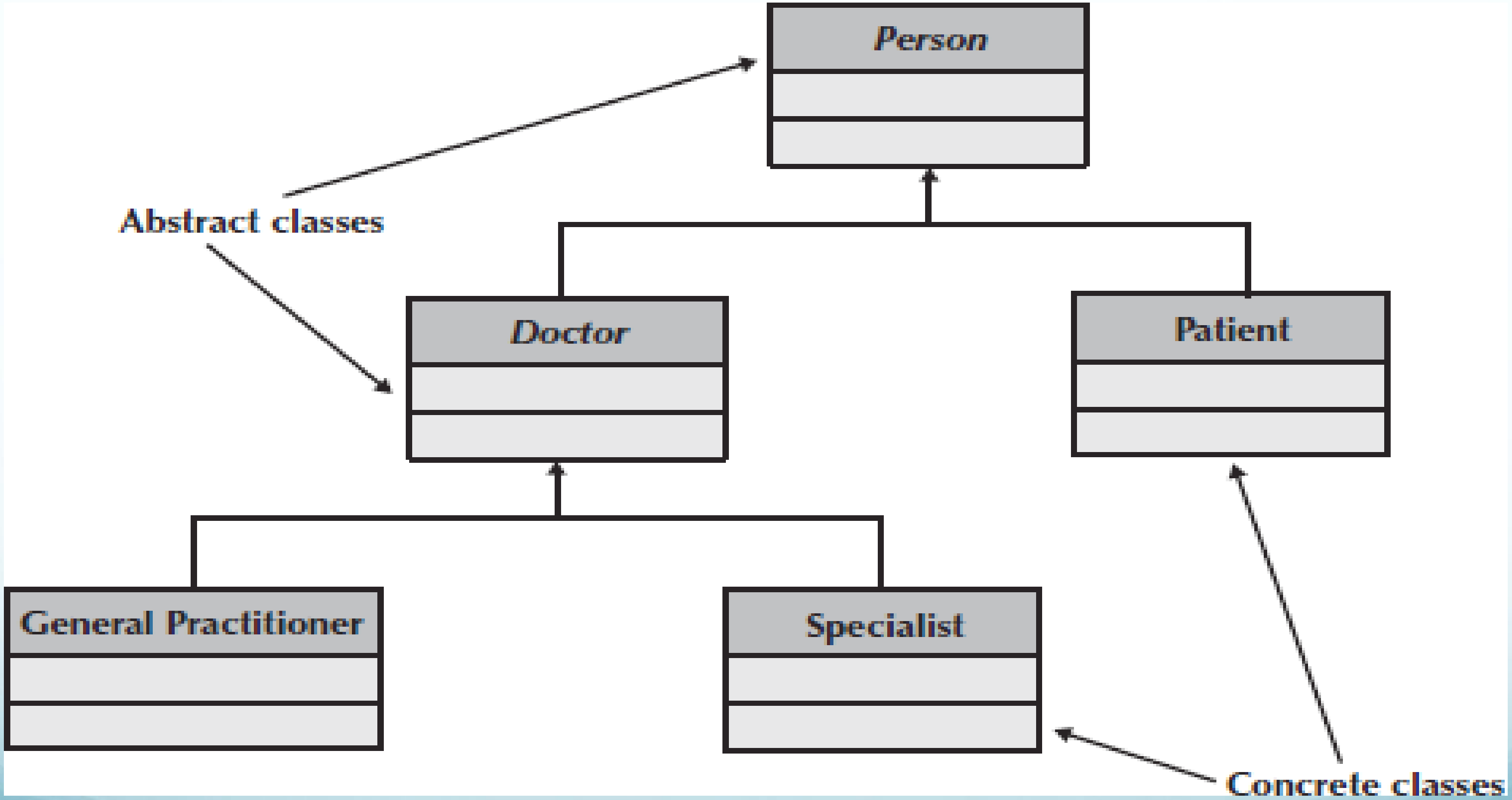
Characteristics of Object-Oriented Systems (cont.)

- Encapsulation & information hiding
  - Encapsulation: combination of process & data
  - Information hiding
  - The method caller is not concerned with how a method is implemented.
  - Therefore it cannot care of if that procedure is changed, so long as it works.
  - Public/private declarations enforce information hiding in the code.

Characteristics of Object-Oriented Systems (cont.)

- Inheritance
  - General classes are created (superclasses)
  - Subclasses can inherit data and methods from a superclass
  - A subclass is a specialized type within the superclass.

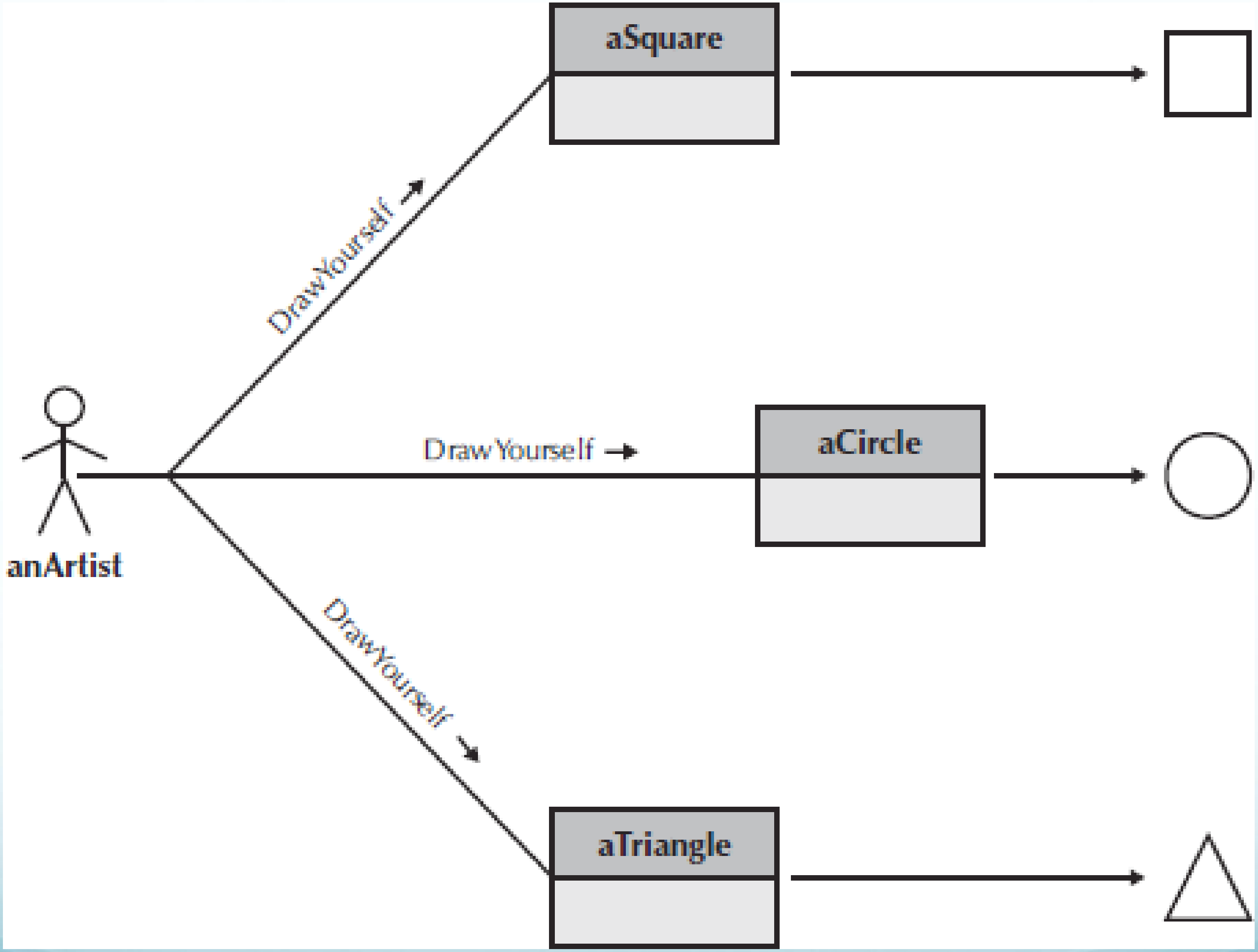
Characteristics of Object-Oriented Systems (cont.)



Characteristics of Object-Oriented Systems (cont.)

- Polymorphism
  - Different object types provide different meanings for the same message.
  - Therefore the meaning of a message (method) depends on the type of object receiving it.
- Dynamic binding
  - Implements polymorphism in a programming language.
  - A variable may hold different types of object at different times.
  - When a method is called on that variable, the code to run is chosen by what object type is held at that moment.

Polymorphism



Dynamic Binding

```
class Shape { }
class Square extends Shape { draw() { } }
class Circle extends Shape { draw() { } }
class Triangle extends Shape { draw() { } }

Shape s;
. . .
s.draw();
```

Which draw() is called?

Object-Oriented Systems Analysis & Design

- Use-case driven
  - Use-cases define the behavior of a system
  - Each use-case focuses on one business process
- Architecture centric
  - Functional (external) view: focuses on the user’s perspective
  - Static (structural) view: focuses on attributes, methods, classes & relationships
  - Dynamic (behavioral) view: focuses on messages between classes and resulting behaviors

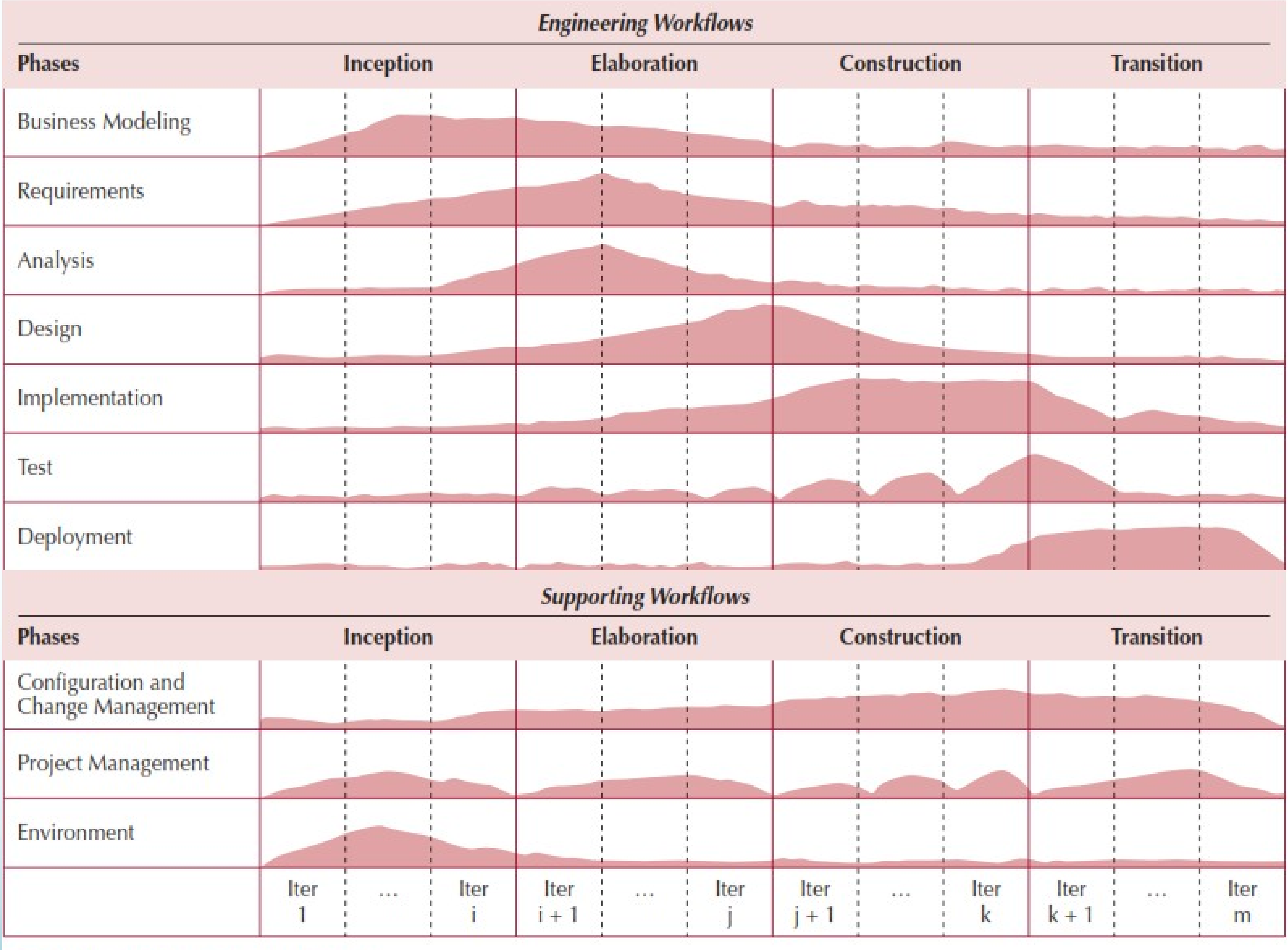
Object-Oriented Systems Analysis & Design (cont.)

- Iterative & incremental
  - Undergoes continuous testing & refinement
  - The analyst understands the system better over time
- Benefits of OOSAD
  - Break a complex system into smaller, more manageable modules
  - Work on modules individually

The Unified Process

- A specific methodology that maps out when and how to use the various UML techniques for object-oriented analysis and design
- A two-dimensional process consisting of phases and workflows
  - Phases are time periods in development
  - Workflows are the tasks that occur in each phase
  - Activities in both phases & workflows will overlap

The Unified Process



Unified Process Phases

- Inception
  - Feasibility analyses performed
  - Workflows vary but focus is on business modeling & requirements gathering
- Elaboration
  - Heavy focus on analysis & design
  - Other workflows may be included
- Construction: Focus on programming (implementation)
- Transition--Focus on testing & deployment

Engineering Workflows

- Business modeling
- Requirements
- Analysis
- Design
- Implementation
- Testing
- Deployment

Supporting Workflows

- Project management
- Configuration and change management
- Environment
- Operations and support*
- Infrastructure management*

* Part of the *enhanced* unified process

Extensions to the Unified Process

- The Unified Process does not include:
  - Staffing
  - Budgeting
  - Contract management
  - Maintenance
  - Operations
  - Support
  - Cross- or inter-project issues

Extensions to the Unified Process (cont.)

- Add a Production Phase to address issues after the product has been deployed
- New Workflows:
  - Operations & Support
  - Infrastructure management
- Modifications to existing workflows:
  - Test workflow
  - Deployment workflow
  - Environment workflow
  - Project Management workflow
  - Configuration & change management workflow

Unified Modeling Language

- Provides a common vocabulary of object-oriented terms and diagramming techniques rich enough to model any systems development project from analysis through implementation
- Version 2.5 has 15 diagrams in 2 major groups:
  - Structure diagrams
  - Behavior diagrams

UML Structure Diagrams

- Represent the data and static relationships in an information system
  - Class
  - Object
  - Package
  - Deployment
  - Component
  - Composite structure

UML Behavior Diagrams

- Depict the dynamic relationships among the instances or objects that represent the business information system
  - Activity
  - Sequence
  - Communication
  - Interaction overview
  - Timing
  - Behavior state machine
  - Protocol state machine,
  - Use-case diagrams

Summary

- All systems development projects follow essentially the same process, called the system development life cycle (SDLC)
- System development methodologies are formalized approaches to implementing SDLCs
- The systems analyst needs a variety of skills and plays a number of different roles
- Object-oriented systems differ from traditional systems

Summary

- Object-Oriented Systems Analysis and Design (OOSAD) uses a use-case-driven, architecture-centric, iterative, and incremental information systems development approach
- The Unified Process is a two-dimensional systems development process described with a set of phases and workflows
- The Unified Modeling Language, or UML, is a standard set of diagramming techniques