Systems Analysis and Design With UML 2.0

An Object-Oriented Approach, Second Edition

Chapter 2: Introduction to Object-Oriented Systems Analysis and Design with the Unified Modeling Language, Version 2.0

Alan Dennis, Barbara Wixom, and David Tegarden

WILEY

Copyright © 2005 John Wiley & Sons, Inc.

WILEY

Unified Modeling Language, Version 2.0

Chapter 2

Objectives

- Understand the basic characteristics of object-oriented systems.
- Be familiar with the Unified Modeling Language (UML), Version 2.0.
- Be familiar with the Unified Process.
- Understand a minimalist approach to object-oriented systems analysis and design.

Basic Characteristics of Object Oriented Systems

- Classes and Objects
- Methods and Messages
- Encapsulation and Information Hiding
- Inheritance
- Polymorphism and Dynamic Binding

Classes and Objects

- Class – Template to define specific instances or objects
- Object – Instantiation of a class
- Attributes – Describes the object
- Behaviors – specify what object can do

Classes and Objects

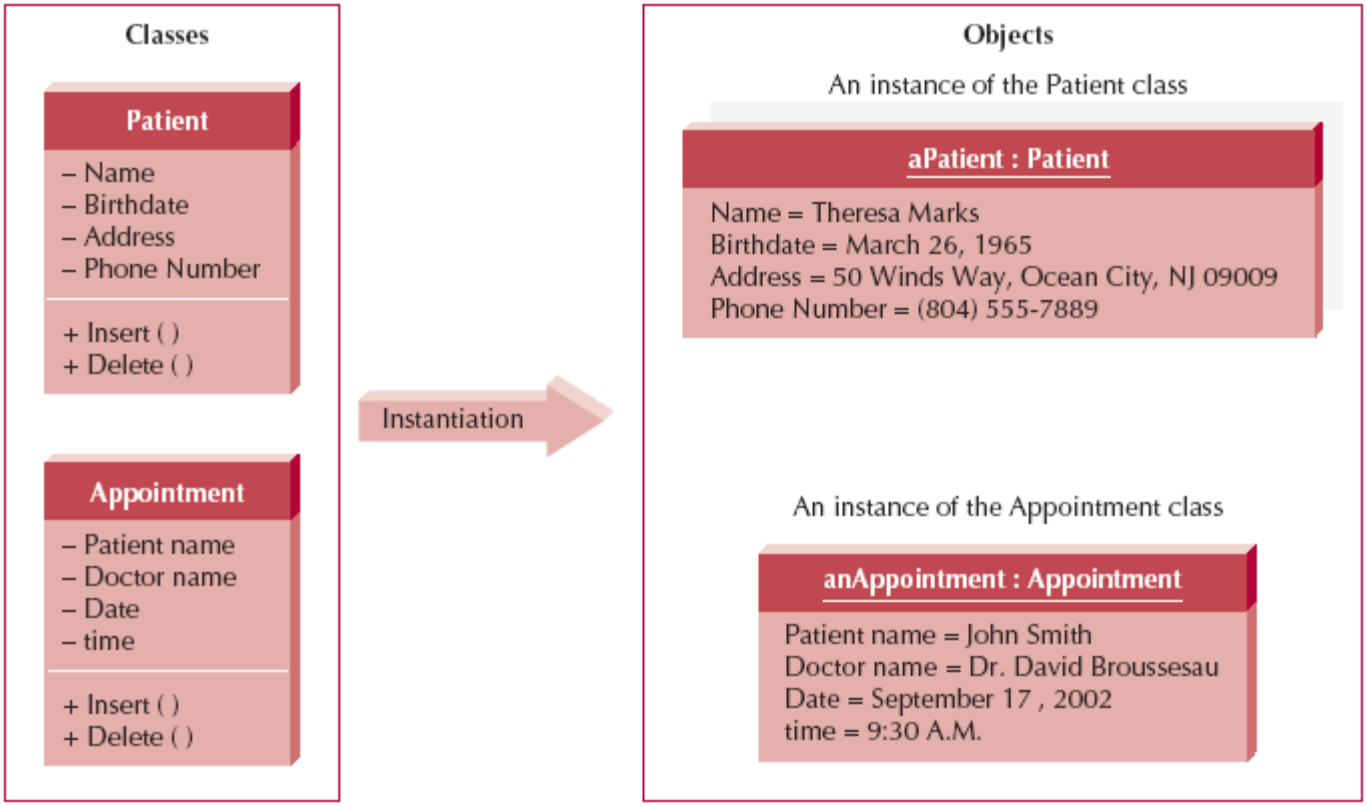


FIGURE 2-1 Classes and Objects

Methods and Messages

- Methods implement an object's behavior
  - Analogous to a function or procedure
- Messages are sent to trigger methods
  - Procedure call from one object to the next

Messages and Methods

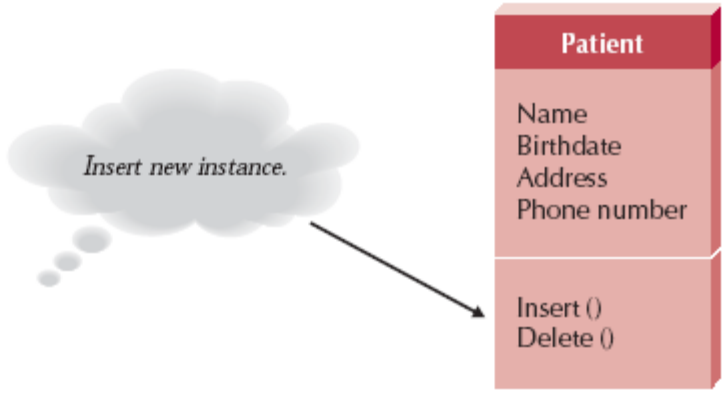


FIGURE 2-2
Messages and Methods

A message is sent to the application.

The object's insert method will respond to the message and insert a new patient instance.

Encapsulation and Information Hiding

- Encapsulation
  - combination of data and process into an entity
- Information Hiding
  - Only the information required to use a software module is published to the user
- Reusability Key
  - Use an object by calling methods

Inheritance

- Superclasses or general classes are at the top of a hierarchy of classes
- Subclasses or specific classes are at the bottom
- Subclasses inherit attributes and methods from classes higher in the hierarchy

Class Hierarchy

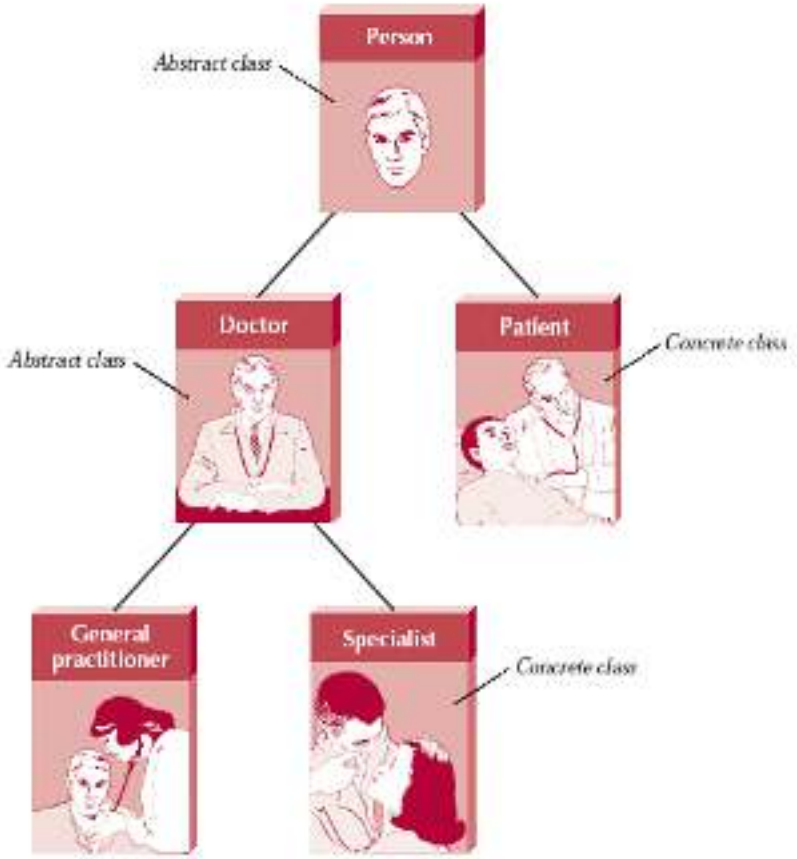


FIGURE 2-3
Class Hierarchy

Inheritance

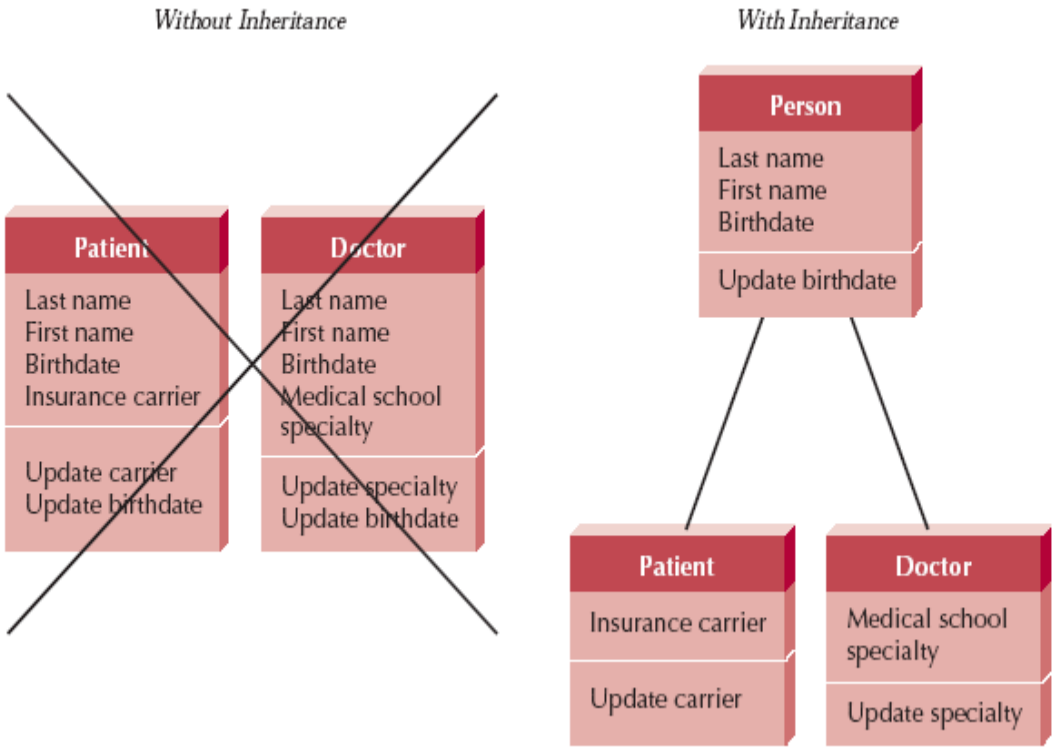


FIGURE 2-4
Inheritance

Polymorphism and Dynamic Binding

- Polymorphism
  - A message can be interpreted differently by different classes of objects
- Dynamic Binding
  - Sometimes called late binding
  - Delays typing or choosing a method for an object until run-time
- Static Binding
  - Type of object determined at compile time

Polymorphish & Encapsulation

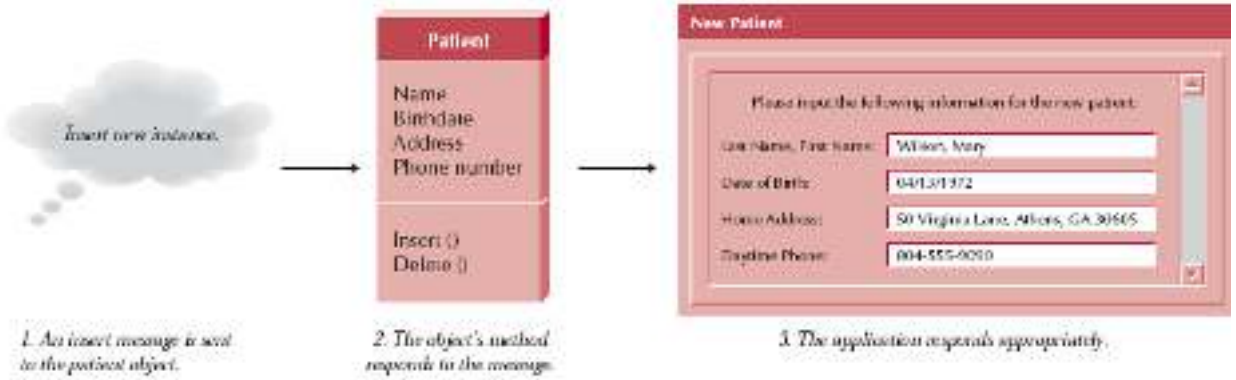


FIGURE 2-5 Polymorphism and Encapsulation

The Unified Modeling Language, Version 2.0

- Structure Diagrams
- Behavior Diagrams
- Extension Mechanisms
- Developers
  - Grady Booch
  - Ivar Jacobson
  - James Rumbaugh

Structure Diagram

- Structure Diagrams include
  - Class
  - Object
  - package
  - Deployment
  - Component
  - Composite structure diagrams

Diagram Name	Used to	Primary Phase
Structure Diagrams		
Class	Illustrate the relationships between classes modeled in the system.	Analysis, Design
Object	Illustrate the relationships between objects modeled in the system. Used when actual instances of the classes will better communicate the model.	Analysis, Design
Package	Group other UML elements together to form higher level constructs.	Analysis, Design, Implementation
Deployment	Show the physical architecture of the system. Can also be used to show software components being deployed onto the physical architecture.	Physical Design, Implementation
Component	illustrate the physical relationships among the software components.	Physical Design, Implementation
Composite Structure	Illustrate the internal structure of a class, i.e., the relationships among the parts of a class.	Analysis, Design
Behavioral Diagrams		
Activity	Illustrate business workflows independent of classes, the flow of activities in a use case, or detailed design of a method.	Analysis, Design
Sequence	Model the behavior of objects within a use case. Focuses on the time-based ordering of an activity.	Analysis, Design
Communication	Model the behavior of objects within a use case. Focuses on the communication among a set of collaborating objects of an activity.	Analysis, Design
Interaction Overview	Illustrate an overview of the flow of control of a process.	Analysis, Design
Timing	Illustrate the interaction that takes place among a set of objects and the state changes in which they go through along a time axis.	Analysis, Design
Behavioral State Machine	Examine the behavior of one class.	Analysis, Design
Protocol State Machine	Illustrates the dependencies among the different interfaces of a class.	Analysis, Design
Use-Case	Capture business requirements for the system and to illustrate the interaction between the system and its environment.	Analysis

FIGURE 2-6 UML 2.0 Diagram Summary

Structure Diagrams

- Class
  - relationship between classes
- Object
  - Relationships between objects
- Package
  - Group UML elements together to form higher level constructs

Structure Diagrams Cont.

- Deployment
  - Shows the physical architecture and software components of system
- Component
  - Physical relationships among software components
- Composite Structure
  - Illustrates internal structure of a class

Activity Diagrams

- Activity
  - Illustrates business workflows
- Sequence
  - Time-based ordering Behavior of objects activities in a use case
- Communication
  - Communication among a set of collaborating objects of an activity
- Interaction Overview Timing
  - Overview of flow of control of a process

State Machines

- Behavioral State Machine
  - Examines behavior of one class
- Protocol State Machine
  - Shows dependencies of different interfaces of a class
- Use-Case
  - Captures business requirements
  - Illustrates interaction between system and environment

Use Case Diagrams

- Captures Business requirements
- Illustrates interaction between a system and its environment
  - Includes end user
  - Any external system that interacts with its information system
- Documents and clarifies requirements of system being modeled

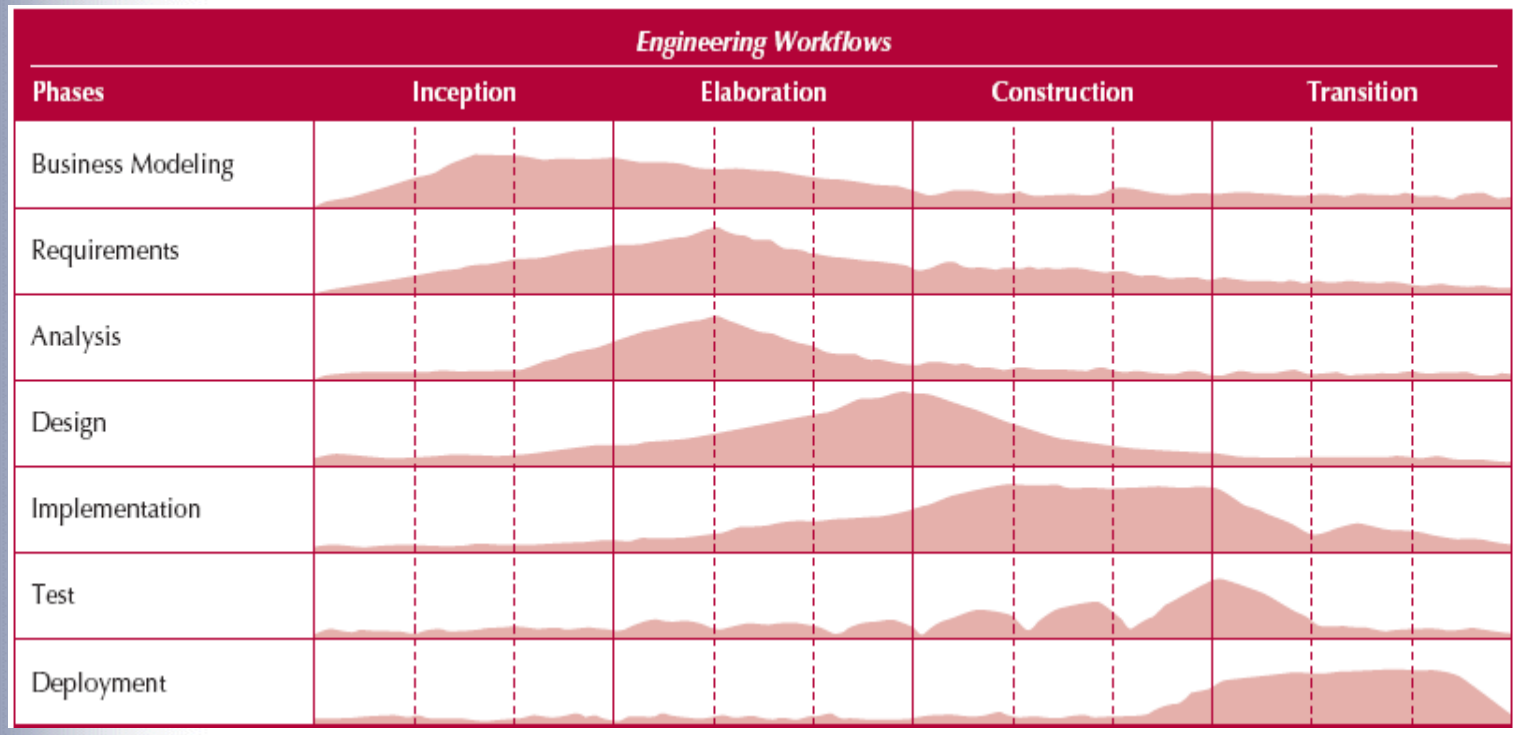
Extension Mechanisms

- Stereotypes
  - Gives ability to incrementally extend UML
- Tagged Values
  - Add new properties to base elements
- Constraints
  - Place restrictions on use of model elements
- Profiles
  - Group model elements into a package

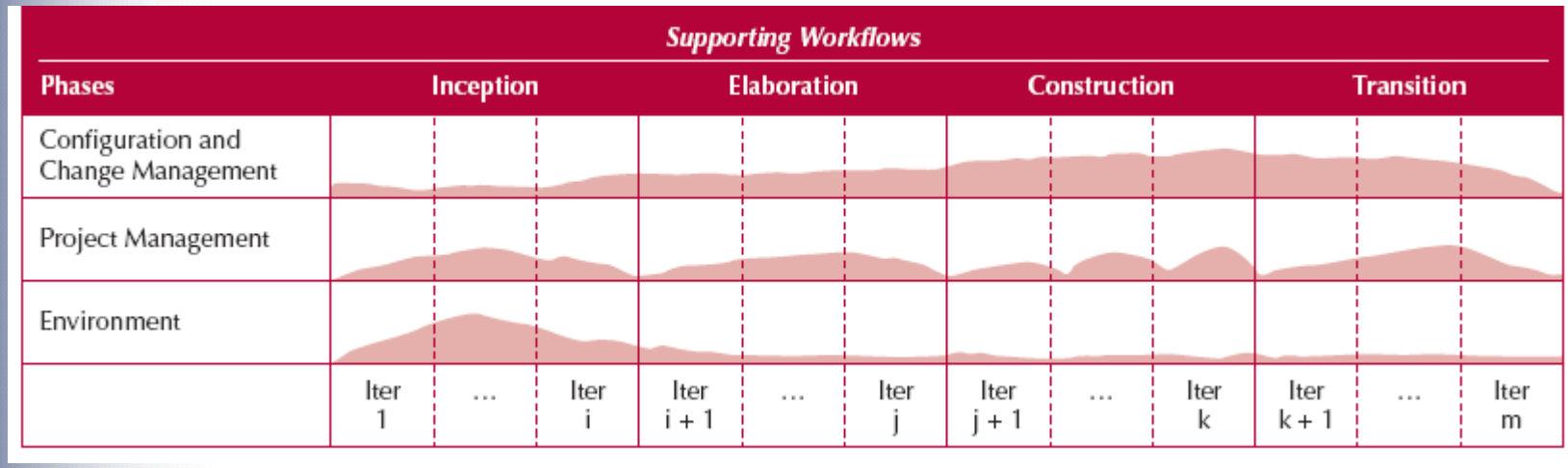
Object Oriented Systems Analysis and Design

- Use-case driven
- Architecture Centric
- Iterative and Incremental
- The Unified Process

Engineering Workflows



Supporting Workflows



A Minimalist Approach

- Benefits of Object-Oriented Systems Analysis and Design
- Extensions of the Unified Process
- The Minimalist Object-Oriented Systems Analysis and Design Approach

Benefits of the Object Approach

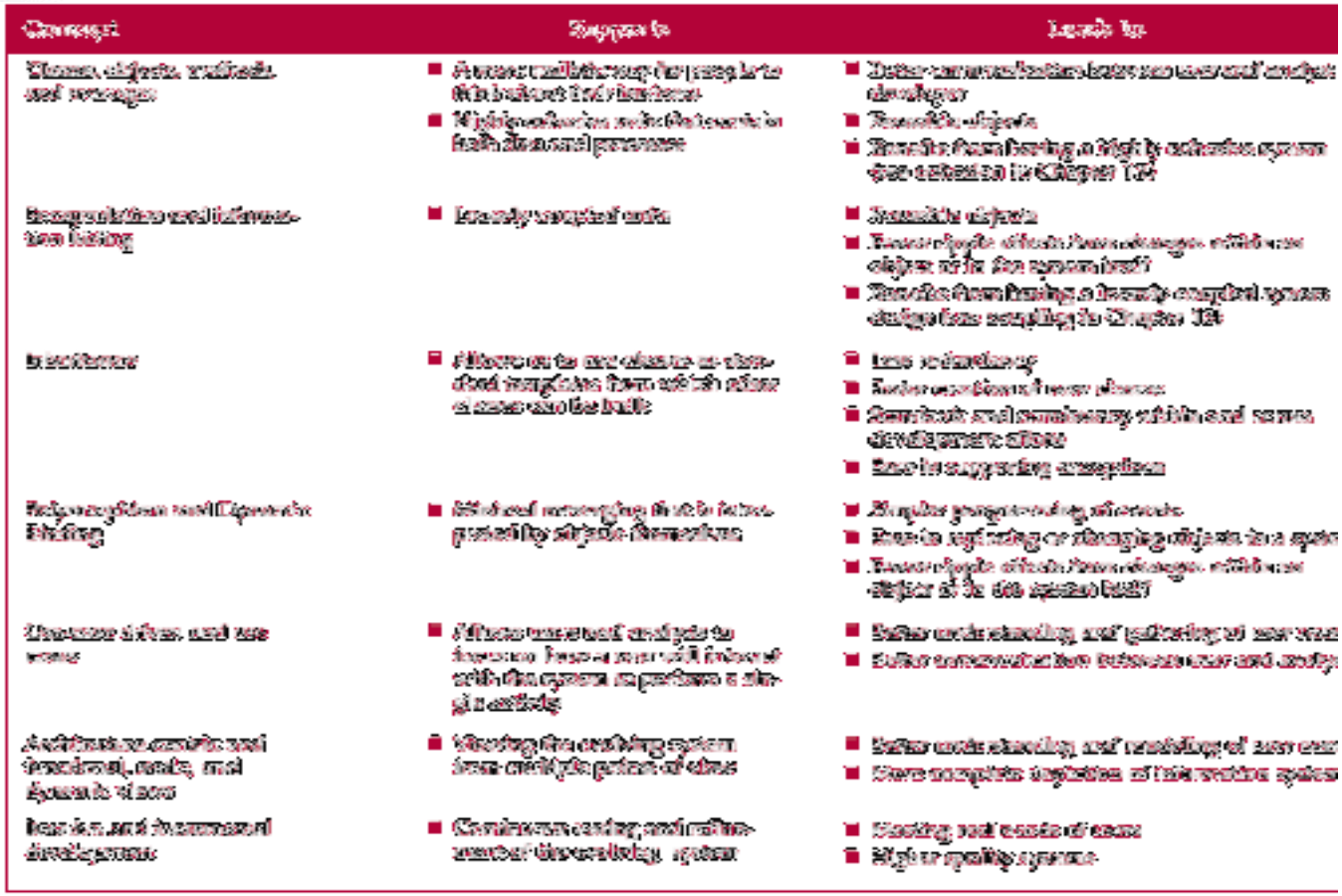


FIGURE 2-5 Benefits of the Object Approach

MOOSAD Approach

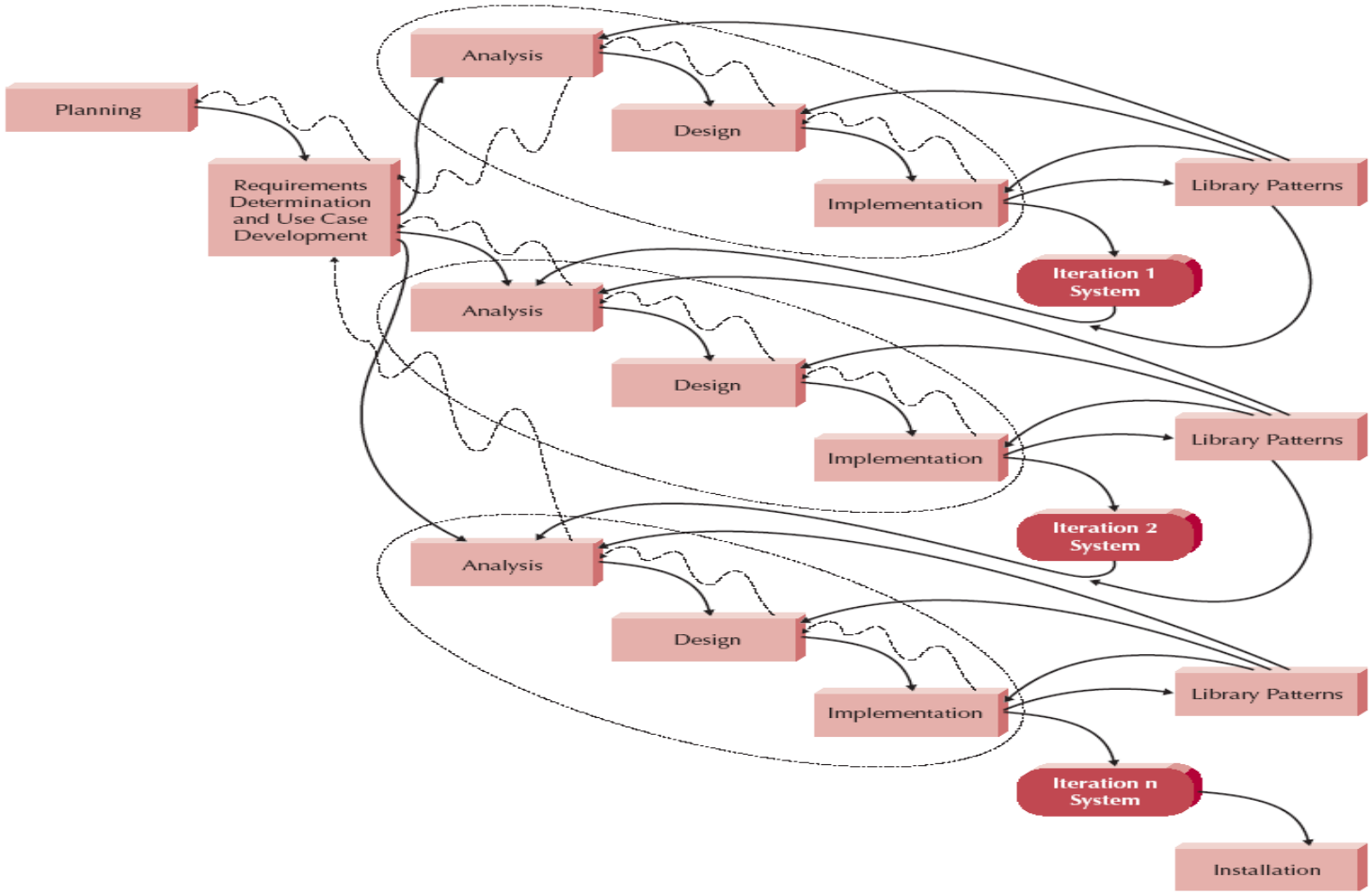


FIGURE 2-10 The Minimalist Object-Oriented Systems Analysis and Design (MOOSAD) Approach

Basic Characteristics of Object Oriented Systems

- Identifying business value
- Analyze feasibility
- Develop workplan
- Staff the project
- Control and direct project
- Requirements determination
- Functional modeling
- Structural modeling
- Behavioral modeling
- Moving on to design

UML Summary

- Class and method design
- Data management layer design
- Human computer interaction layer design
- Physical architecture layer design
- Construction
- Installation
- Operations and support

Summary

- Basic characteristics of an object oriented system
- Unified modeling system
- Object oriented Systems Analysis and Design
- Minimalist approach to Object oriented systems analysis and design with UML