西安电子科技大学
XIDIAN UNIVERSITY

A

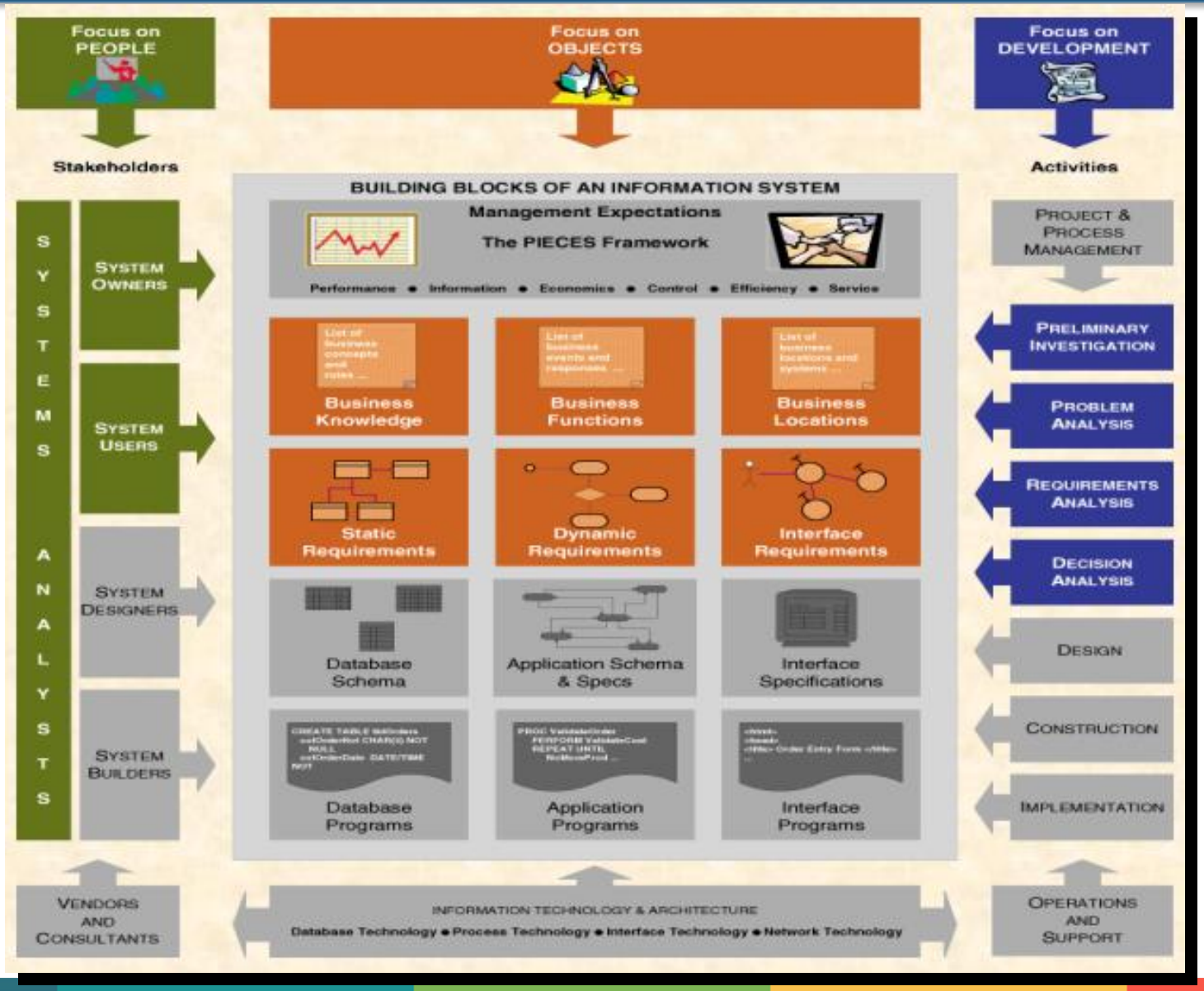
系统分析与设计
（System Analysis and Design）

Object-oriented Analysis and Modeling

Content Structure

- An Introduction to Object Modeling
- System Concepts for Object Modeling
  - 面向对象的基本概念。
- The UML Diagrams
- The Process of Object Modeling

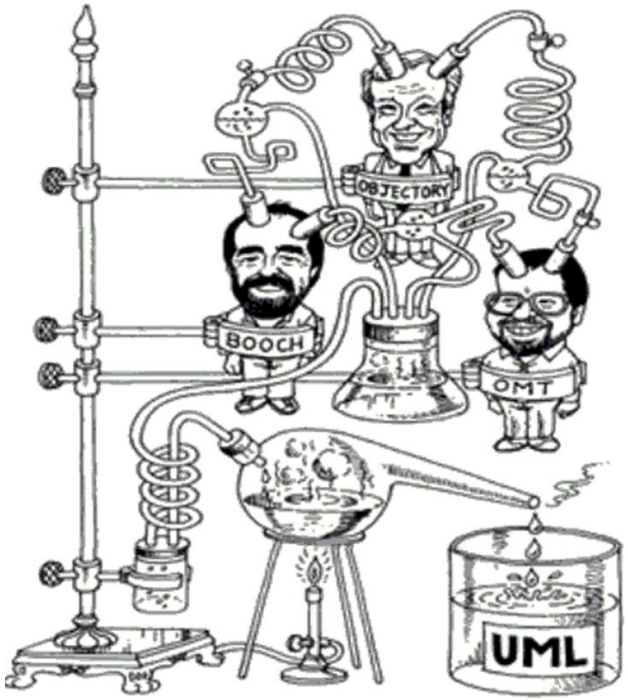
Chapter Map


Focus on PEOPLE
Focus on OBJECTS
Focus on DEVELOPMENT
Stakeholders
Activities
BUILDING BLOCKS OF AN INFORMATION SYSTEM
Management Expectations
The PIECES Framework
Performance • Information • Economics • Control • Efficiency • Service
SYSTEMS ANALYSTS
System Owners
System Users
System Designers
System Builders
Business Knowledge
Business Functions
Business Locations
Static Requirements
Dynamic Requirements
Interface Requirements
Database Schema
Application Schema & Specs
Interface Specifications
Database Programs
Application Programs
Interface Programs
Project & Process Management
Preliminary Investigation
Problem Analysis
Requirements Analysis
Decision Analysis
Design
Construction
Implementation
Vendors and Consultants
INFORMATION TECHNOLOGY & ARCHITECTURE
Database Technology • Process Technology • Interface Technology • Network Technology
Operations and Support

An Introduction to Object Modeling

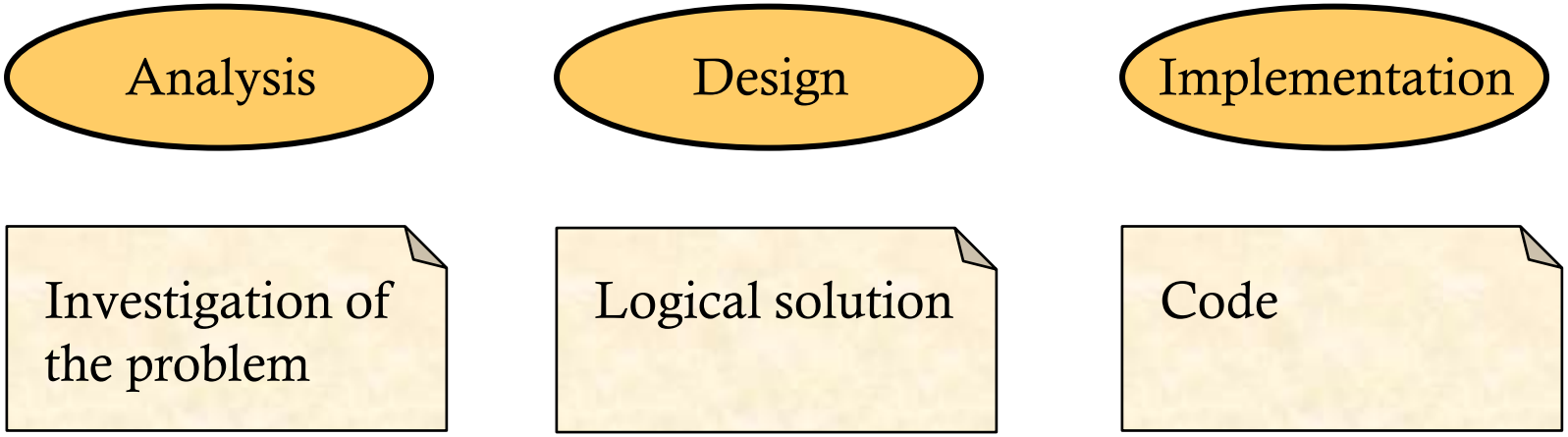
Introduction to Object Modeling

- **Object-oriented analysis 面向对象分析** (OOA) techniques are used to (1) study existing objects to see if they can be reused or adapted for new uses, and (2) define new or modified objects that will be combined with existing objects into a useful business computing application.
- **Object Modeling 对象建模** is a technique for identifying objects within the systems environment and the relationships between those objects.
  - Booch - Grady Booch
  - OMT - James Rumbaugh
  - OOSE - Ivar Jacobson



Other Related Definitions

- Object-oriented analysis
  - Investigation that is object-centric.
- Object-oriented design
  - Solution in terms of interacting software objects
- Object-oriented programming
  - Coding in an object-oriented programming language.



Other Related Definitions

- Requirements Analysis: *discover and express* requirements in *use cases*. Use Case is a textual description of a business process in the system.
- Domain Analysis: *develop* a *conceptual model* of the problem domain. This includes the things, the concepts, as well as the various roles people may take and the relationships between them.
- Design - Assignment of Responsibilities: allocate tasks to software objects as well as roles people take, illustrated in *interaction diagrams* and *logical class diagrams*.

Introduction to the UML

- The **Unified Modeling Language 统一建模语言** (UML) is a set of modeling conventions that is used to specify or describe a software system in terms of objects.
  - The UML does not prescribe a method for developing systems—only a notation that is now widely accepted as a standard for object modeling.

What is UML?

- 1994年10月，Rational公司的Booch和Rumbaugh决定将其Booch方法和OMT方法综合成一个新的建模语言，并于1995年10月。
- 1995年秋季，Jacobson及其OOSE方法加入Rational公司，决定将OOSE方法与Unified Method进行综合，更名为UML，并分别于1996年6月和10月公布了UML 0.9和UML 0.91。
- 1996年，DEC、HP、I-Logix、 Itellicorp、 IBM、ICON、MCI、Microsoft、Oracle、Rational、TI、Unisys发起成立了UML成员协会，于1997年1月推出了UML 1.0，并向OMG申请为一种标准语言。
- 1997年9月产生了UML 1.1，11月被OMG正式采纳。
- 1999年6月，OMG发布了UML 1.3。
- 2001年9月，OMG发布了UML 1.4。
- 2005年，UML2.0；2010年，UML2.3；2012年，UML2.5。

System Concepts for Object Modeling

Objects, Attributes, & Instances

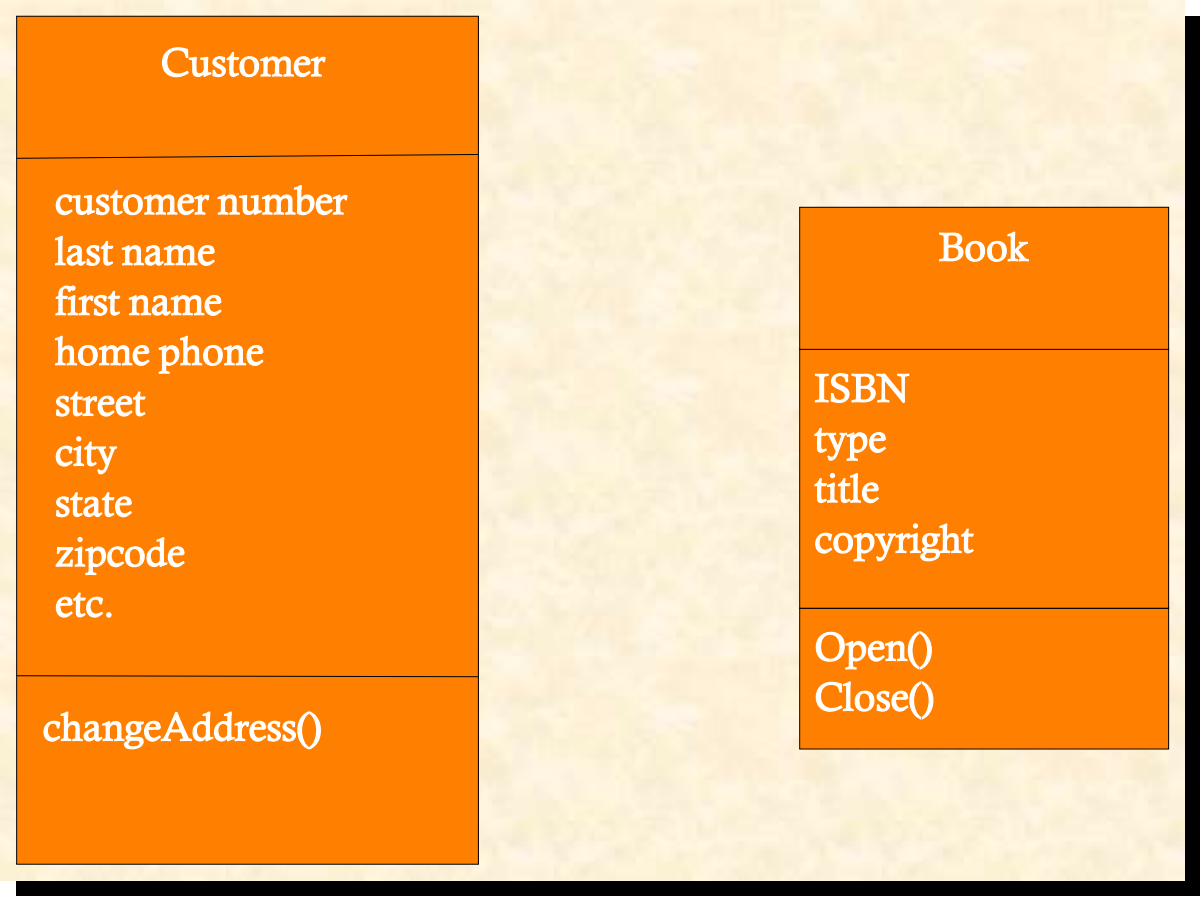
- An object is something that is or is capable of being seen, touched, or otherwise sensed, and about which users store data and associate behavior.
- Attributes are the data that represent characteristics of interest about an object.
- An instance (or *object instance*) of an object consists of the values for the attributes that describe a specific person, place, thing, or event.

Methods & Encapsulation

- Behavior refers to those things that the object can do and which correspond to functions that act on the object's data (or attributes).
  - In object-oriented circles, an object's behavior is commonly referred to as a method or service.
- Encapsulation is the packaging of several items together into one unit (also referred to as information hiding).

Classes

- A class is a set of objects that share common attributes and behavior.



Inheritance

- Inheritance means that methods and/or attributes defined in an object class can be inherited or reused by another object class.

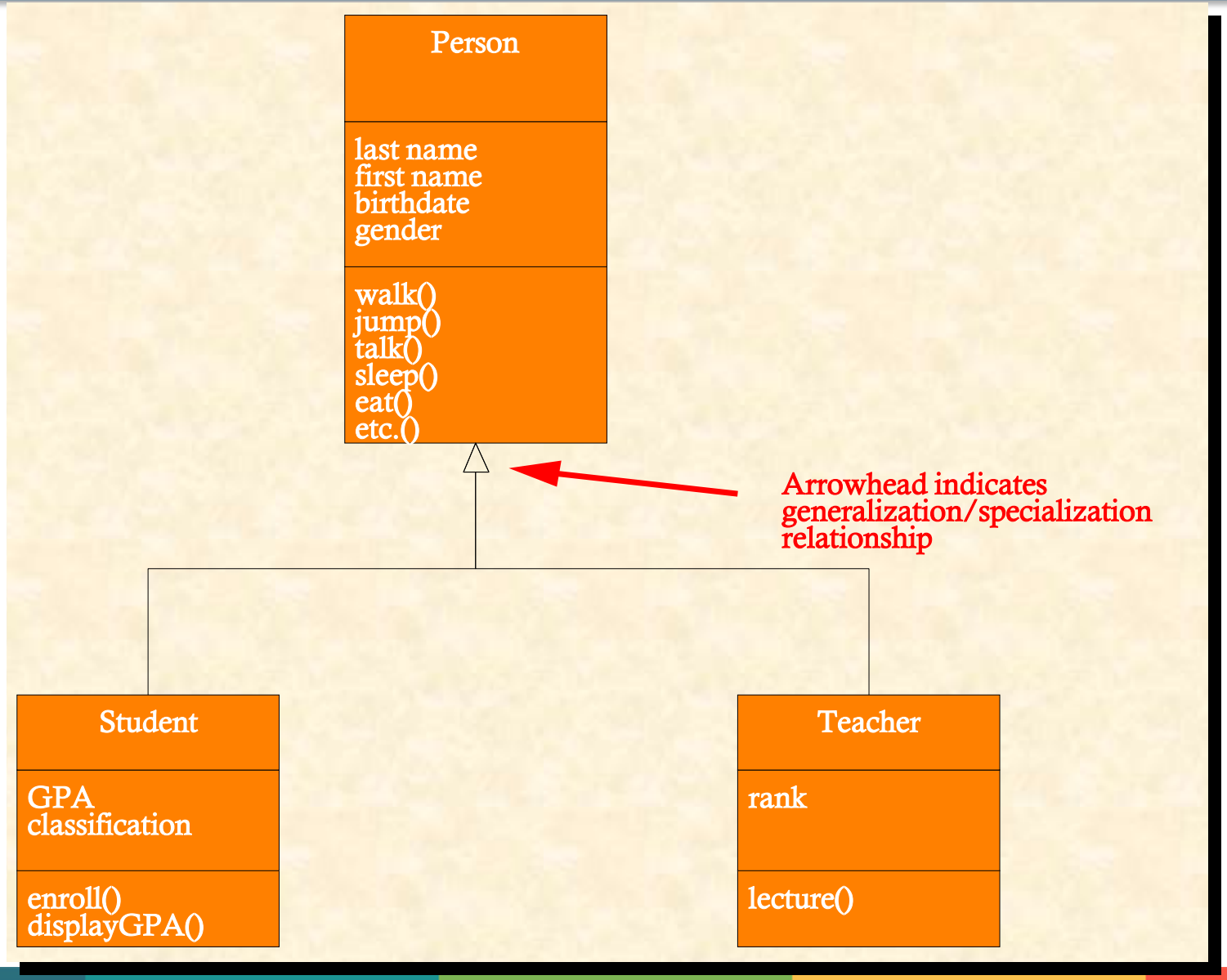
Generalization/Specialization

- Generalization/specialization is a technique wherein the attributes and behaviors that are common to several types of object classes are grouped into their own class, called a supertype. The attributes and methods of the supertype object class are then inherited by those object classes.

Supertypes & Subtypes

- A class supertype is an object class whose instances store attributes that are common to one or more class subtypes of the object.
- A class subtype is an object class whose instances inherit some common attributes from a class supertype, and then add other attributes that are unique to an instance of the subtype.

UML Representation of Generalization/Specialization



Object/Class Relationships

- An object/class relationship is a natural business association that exists between one or more objects/classes.
  - I prefer to call it association in terms of the UML terminology.

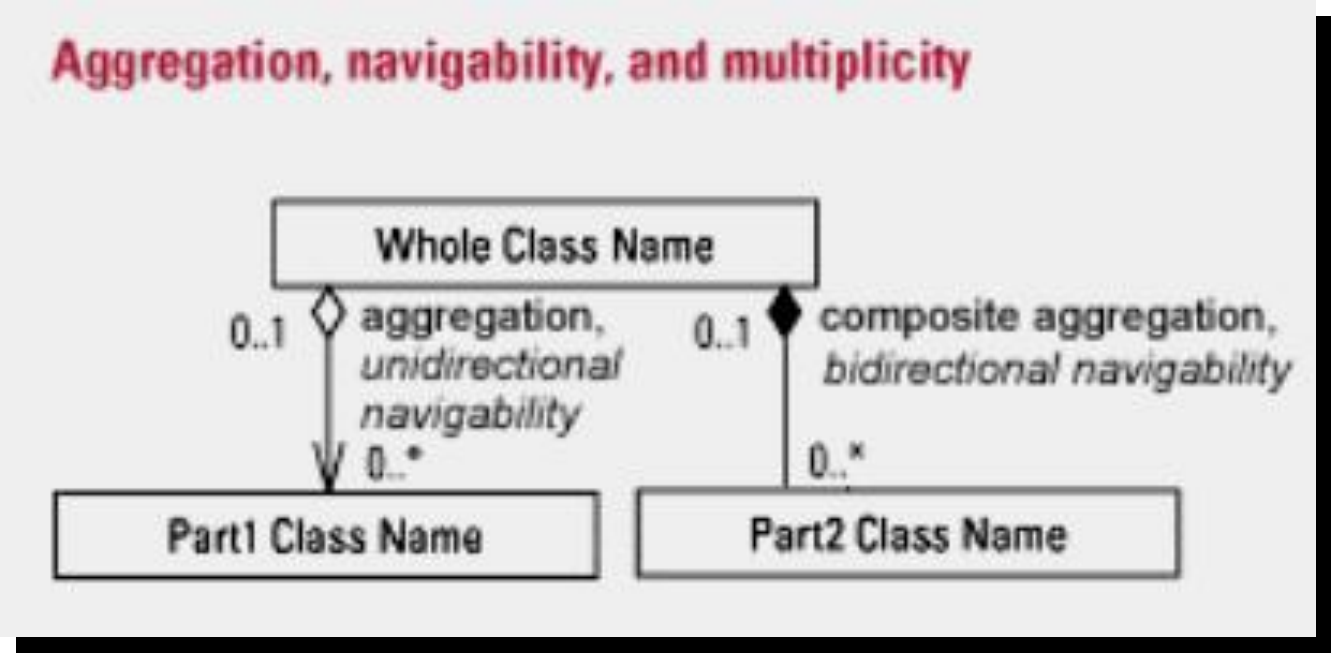
Multiplicity

- **Multiplicity**（重复度） defines how many instances of one object/class can be associated with one instance of another object/class.
  - Recall Chapter 7: Cardinality（基数） defines the minimum and maximum number of occurrences of one entity that may be related to a single occurrence of the other entity.

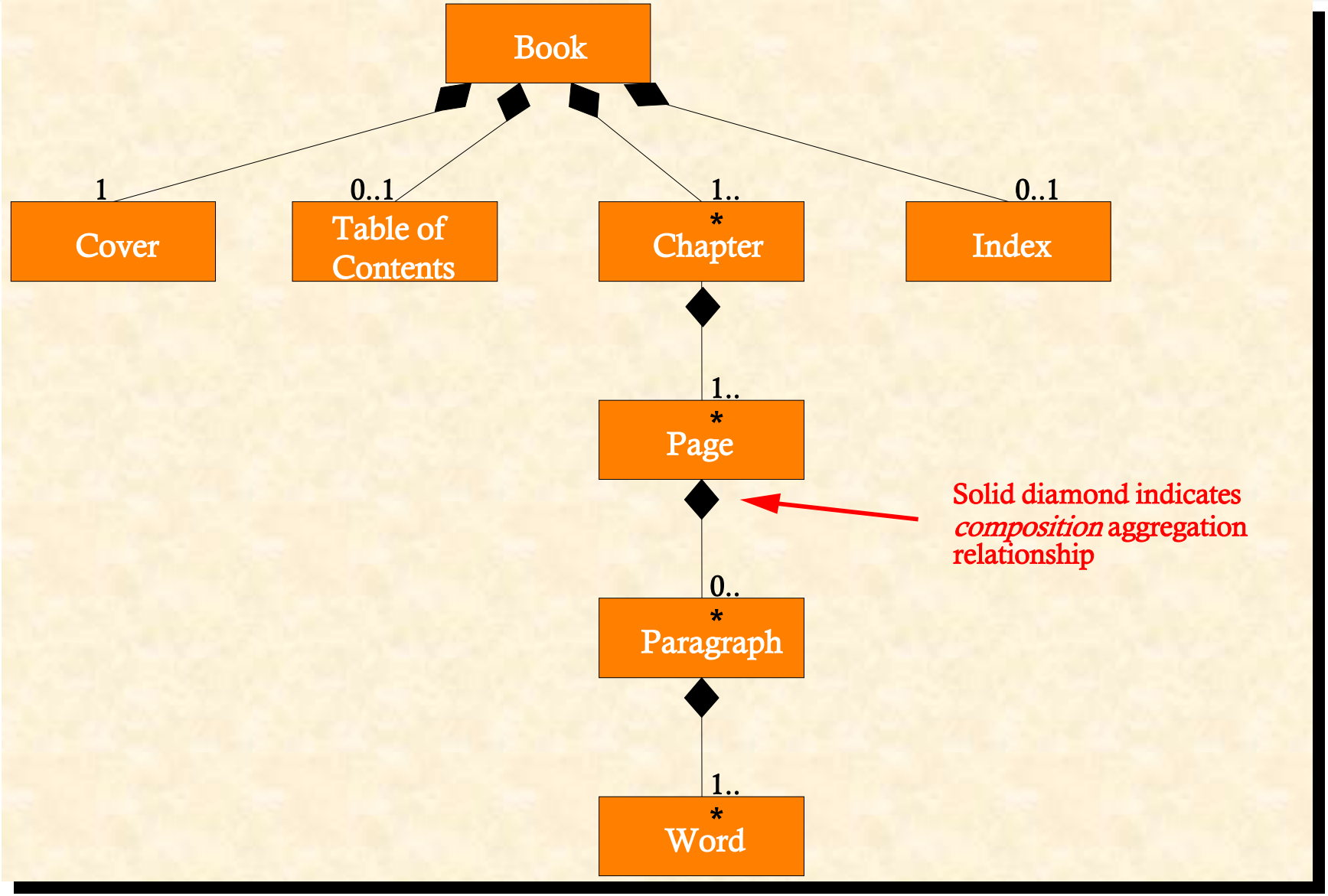
Sample UML Multiplicity Notations

Multiplicity	UML Multiplicity Notation	Association with Multiplicity	Association Meaning
Exactly 1	**1** **or** ***leave blank***	Employee — Works for 1 — Department Employee — Works for — Department	An employee works for one and only one department.
Zero or one	**0..1**	Employee — Has 0..1 — Spouse	An employee has either one or no spouse. 配偶
Zero or more	**0..*** **or** *****	Customer — Makes 0..* — Payment Customer — Makes * — Payment	A customer can make no payment up to many payments.
One or more	**1..***	University — Offers 1..* — Course	A niversity offers at least 1 course up to many courses.
Specific range	**7..9**	Team — Has scheduled 7..9 — Game	A team has either 7, 8, or 9 games scheduled

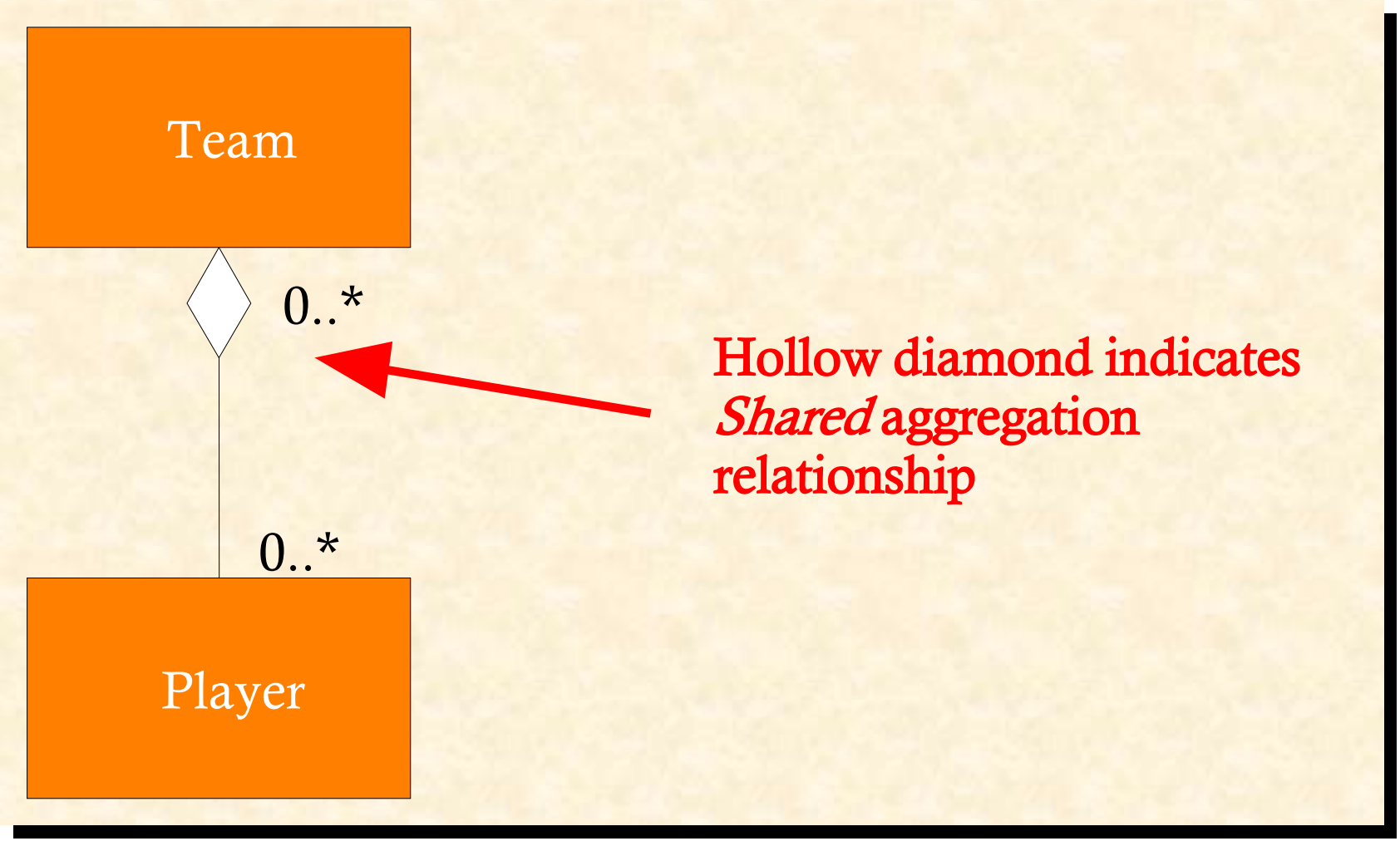
Aggregation in UML



UML Example of a Composite Aggregation Relationship

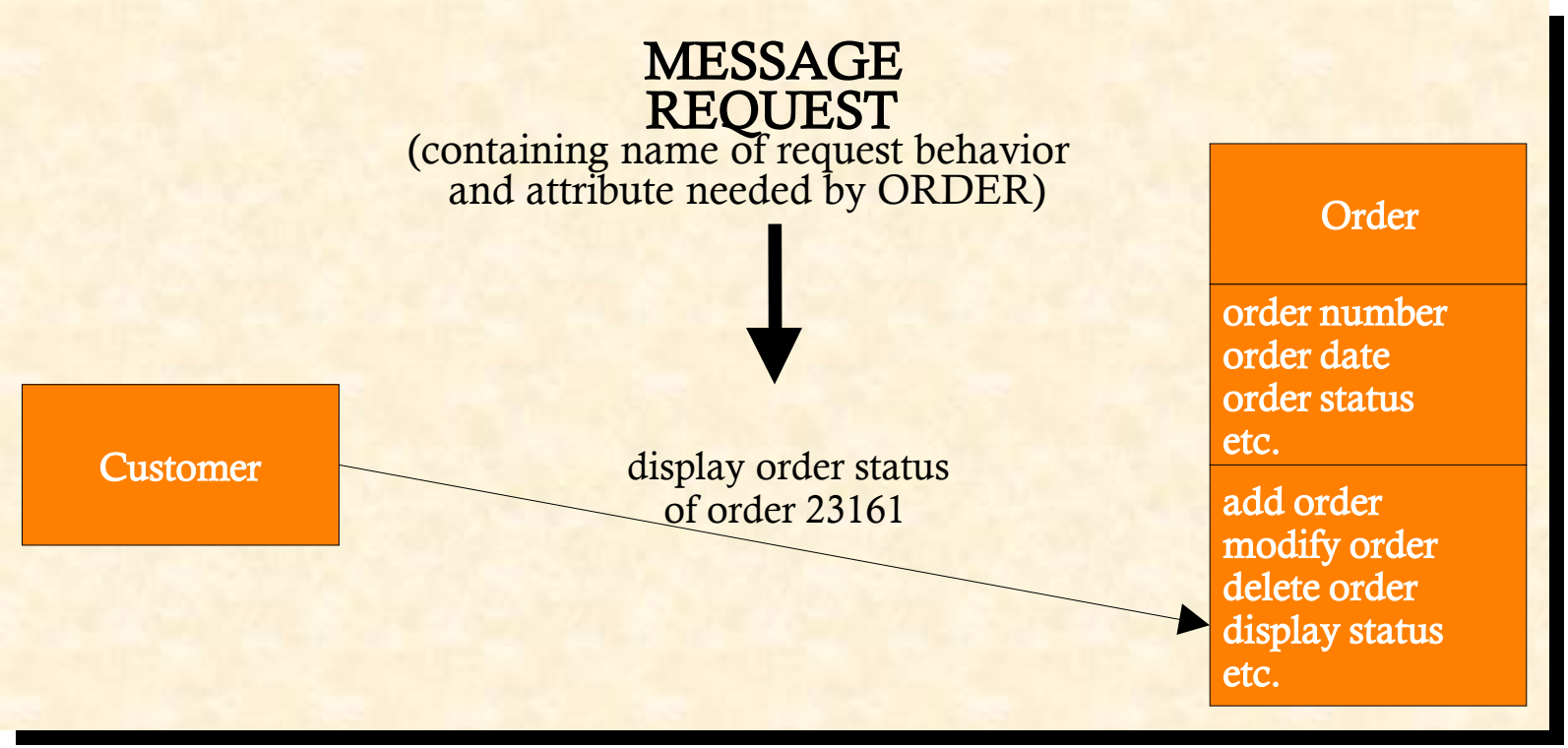


UML Example of a Shared Aggregation Relationship



Messages

- A message is passed when one object invokes one or more of another object's methods (behaviors) to request information or some action.
  - The message is different from the message passing



Polymorphism

- Polymorphism means “many forms.” Applied to object-oriented techniques, it means that a behavior may be completed differently for different objects/classes.
- Here it just say the overriding（重置）, e.g., the virtual function mechanism in C++, not all kinds of polymorphism.

The UML Diagrams

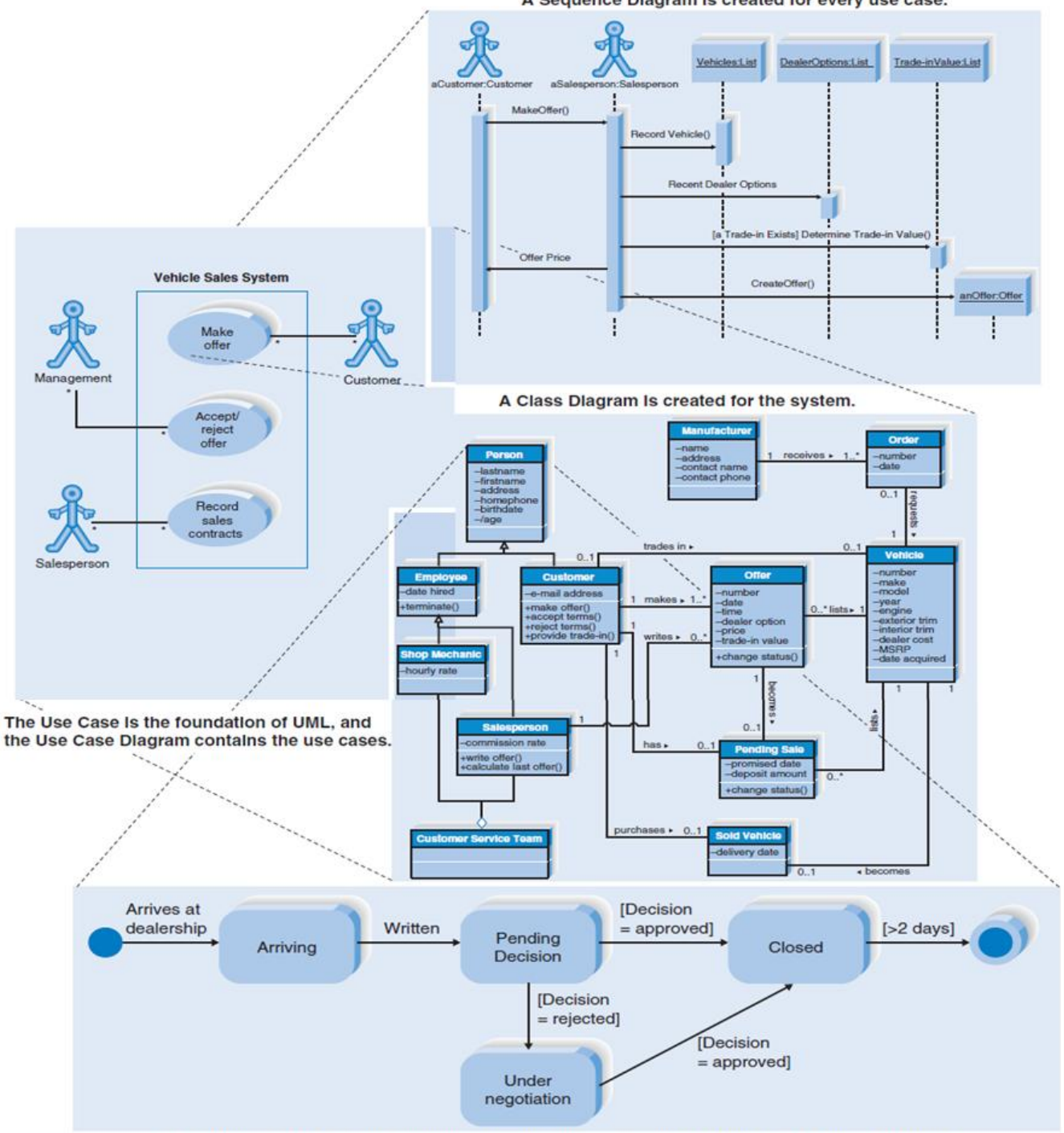
UML Diagrams

- Use Case diagram（用例图） *outer system & user*
- Class diagram（类图） *static structure*
- Object diagram（对象图） *valued snapshot*
- Sequence diagram（序列图） *interaction*
- Collaboration diagram（协同图） *net structure*
- State diagram（状态图） *state transition*
- Activity diagram（活动图） *perform result*
- Component diagram（组件图） *modules*
- Deployment diagram（部署图） *physical structure*

UML 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.	
	Function 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 work flows 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 that they go through along a time axis.	Analysis, Design
Behavioral State Machine	Examine the behavior of one class.	Analysis, Design
Protocol State Machine	Illustrate 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

UML


A Sequence Diagram is created for every use case.
aCustomer:Customer
aSalesperson:Salesperson
Vehicles:List
DealerOptions:List
Trade-inValue:List
MakeOffer()
Record Vehicle()
Recent Dealer Options
[a Trade-in Exists] Determine Trade-in Value()
Offer Price
CreateOffer()
anOffer:Offer
Vehicle Sales System
Management
Make offer
Customer
Accept/ reject offer
Record sales contracts
Salesperson
A Class Diagram is created for the system.
Manufacturer
–name
–address
–contact name
–contact phone
receives
Order
–number
–date
requests
Person
–lastname
–firstname
–address
–homephone
–birthdate
–/age
trades in
Vehicle
–number
–make
–model
–year
–engine
–exterior trim
–interior trim
–dealer cost
–MSRP
–date acquired
Employee
–date hired
+terminate()
Customer
–e-mail address
+make offer()
+accept terms()
+reject terms()
+provide trade-in()
makes
Offer
–number
–date
–time
–dealer option
–price
–trade-in value
+change status()
lists
Shop Mechanic
–hourly rate
writes
becomes
Salesperson
–commission rate
+write offer()
+calculate last offer()
has
Pending Sale
–promised date
–deposit amount
+change status()
The Use Case is the foundation of UML, and the Use Case Diagram contains the use cases.
Customer Service Team
purchases
Sold Vehicle
–delivery date
becomes
Arrives at dealership
Arriving
Written
Pending Decision
[Decision = approved]
Closed
[>2 days]
[Decision = rejected]
Under negotiation
[Decision = approved]
A Behavioral State Machine Diagram is created for every complex class on the Class Diagram.

The Process of Object Modeling

Object-Oriented Analysis General Activities

- Modeling the functions（业务功能） of the system.
- Finding and identifying the business objects.
- Organizing the objects and identifying their relationships.
- Modeling the behavior of the objects.

Use Case Modeling

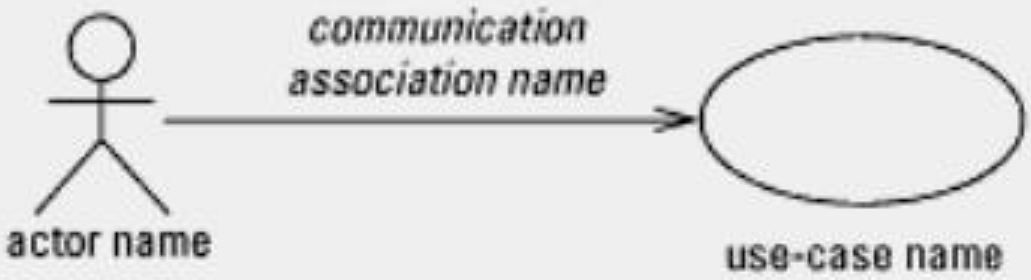
- **Use case modeling** is the process of modeling a system's functions in terms of business events, who initiated the events, and how the system responds to the events.
- A **use case** is a behaviorally related sequence of steps (a scenario（剧情）), both automated and manual, for the purpose of completing a single business task.
- An **actor** represents anything that needs to interact with the system to exchange information. An actor is a user, a role, which could be an external system as well as a person.

Use Case Diagram

USE-CASE DIAGRAM

Shows the system's use cases and which actors interact with them

Actor, use case, and association



Temporal Event Use Cases

- A **temporal event**（时态事件） is a system event that is triggered by time.
  - The actor of a temporal event use case is *time*.

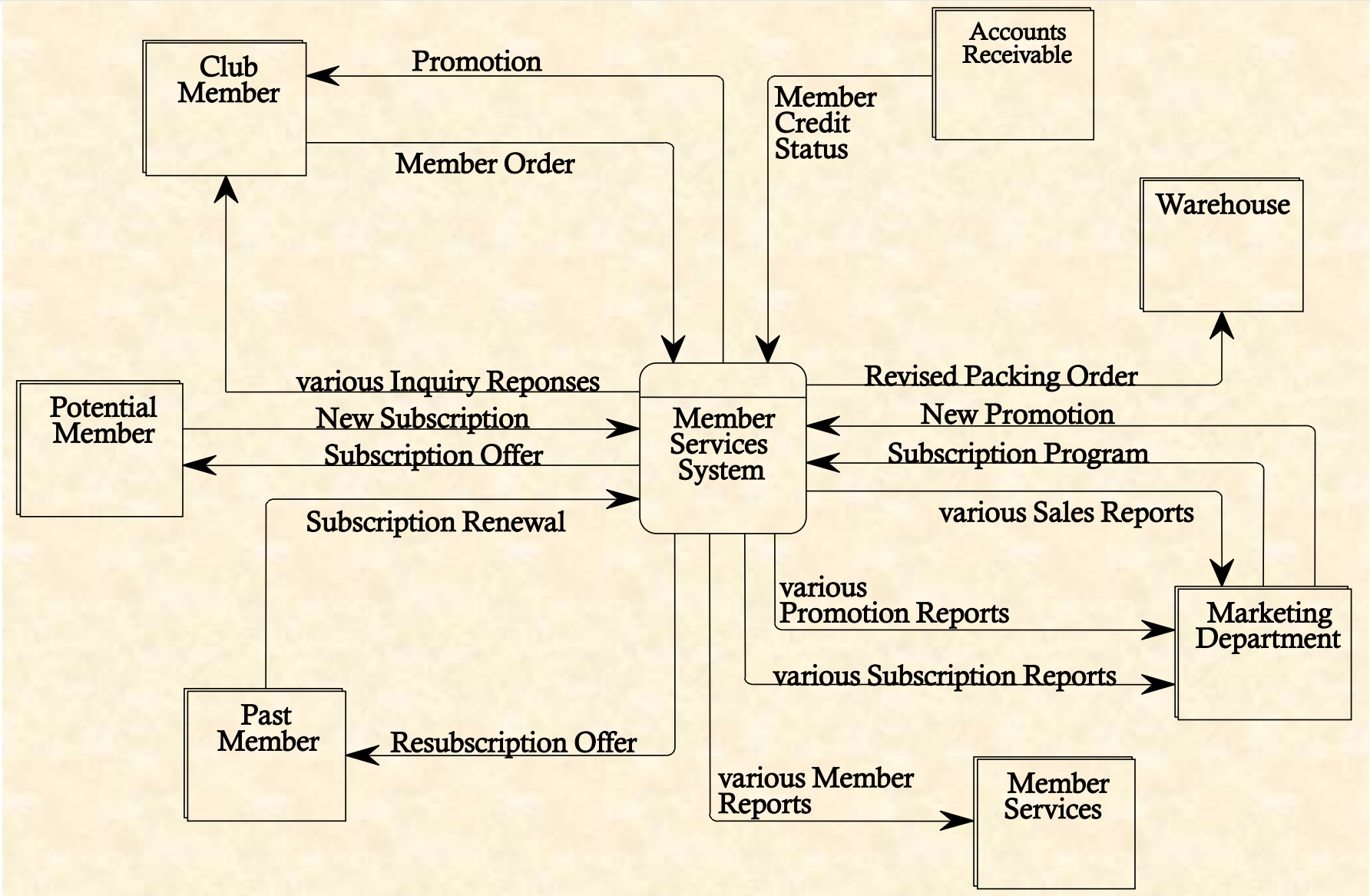
Use Case Modeling Benefits

- As a basis to help identify objects and their high-level relationships and responsibilities.
- A view of system behavior from an external person's viewpoint.
- An effective tool for validating requirements.
- An effective communication tool.
- As a basis for a test plan.
- As a basis for a user's manual.

Use Case Modeling Process

- Step 1: Identifying any additional actors and use cases.
- Step 2: Constructing a use case model.
- Step 3: Document the use case course（过程） of events. (Recall Chapter6)
- Step 4: Define the analysis use cases (each use case will be refined to including more information in order to specify the system functionality in detail).

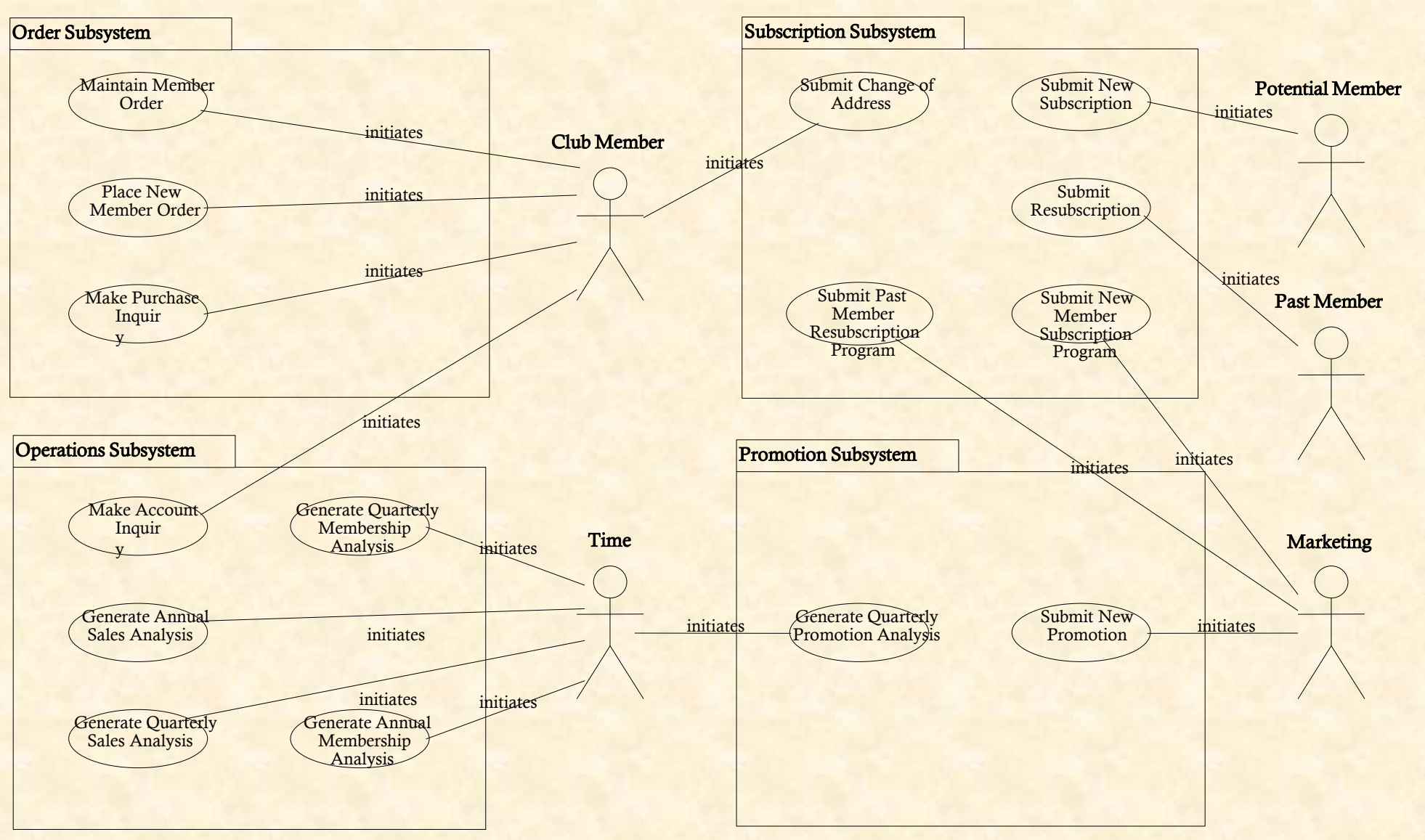
Member Services System Context Model Diagram



Step 1: Identifying Actors and Use Cases

Actor	Use Case Name	Use Case Description
Potential Member	SUBMIT NEW SUBSCRIPTION	Potential member joins the club by subscribing. ("Take anu 12 CDs for one penny and agree to buy 4 more at regular club prices within two years.")
Club Member	PLACE NEW MEMBER ORDER	Club member places order.
Club Member	MAKE ACCOUNT INQUIRY	Club member wants to examine his or her account history. (90-day time limit)
Club Member	MAKE PURCHASE INQUIRY	Club member inquires about his/her purchase history. (three-year time limit)
Club Member	MAINTAIN MEMBER ORDER	Club member wants to revise an order or cancel an order.
Club Member	SUBMIT CHANGE OF ADDRESS	Club member changes address. (including e-mail and privacy code)
Past Member	SUBMIT RESUBSCRIPTION	Past member rejoins the club by resubscribing.
Marketing	SUBMIT NEW MEMBER SUBSCRIPTION PROGRAM	Marketing establishes a new membership resubscription plan to entice new members.
Marketing	SUBMIT PAST MEMBER RESUBSCRIPTION PROGRAM	Marketing establishes a new membership resubscription plan to lure back former members.
Marketing	SUBMIT NEW PROMOTION	Marketing initiates a promotion. (Note: A promotion features specific titles, usually new, that company is trying to sell at a special price. These promotions are integrated into a catalog sent (or communicated) to all members.)
Time	GENERATE QUARTERLY PROMOTION ANALYSIS	Print quarterly promotion analysis report.
Time	GENERATE QUARTERLY SALES ANALYSIS	Print annual sales analysis report.
Time	GENERATE QUARTERLY MEMBERSHIP ANALYSIS	Print annual membership analysis report.
Time	GENERATE ANNUAL SALES ANALYSIS	Print annual sales analysis report.
Time	GENERATE ANNUAL MEMBERSHIP ANALYSIS	Print annual membership analysis report.

Step 2: Constructing a Use Case Model Diagram



Step 3: Documenting the Use Case Typical Course

Author: S. Shepard Date: 10/05/2000

Use Case Name:	Submit New Member Order	
Actor(s):	Member	
Description:	This use case describes the process of a member submitting an order for SoundStage products. On completion, the member will be sent a notification that the order was accepted.	
References	MSS-1.0 1	
Typical Course of Events: 2	**Actor Action** **Step 1**: This use case is initiated when a member submits an order to be processed **Step 7**: This use case concludes when the member receives the order confirmation notice.	**System response** **Step 2**: The member's personal information such as address is validated against what is currently recorded in member services. **Step 3**: The member's credit status is checked with Accounts Receivable to make sure no payments are outstanding. **Step 4**: For each product being ordered, validate the product number and then check the availability in inventory and record the ordered product information. **Step 5**: Create a picking ticket for the member order containing all ordered products that are available and route it to the warehouse for processing. **Step 6**: Generate an order confirmation notice indicating the status of the order and send it to the member.

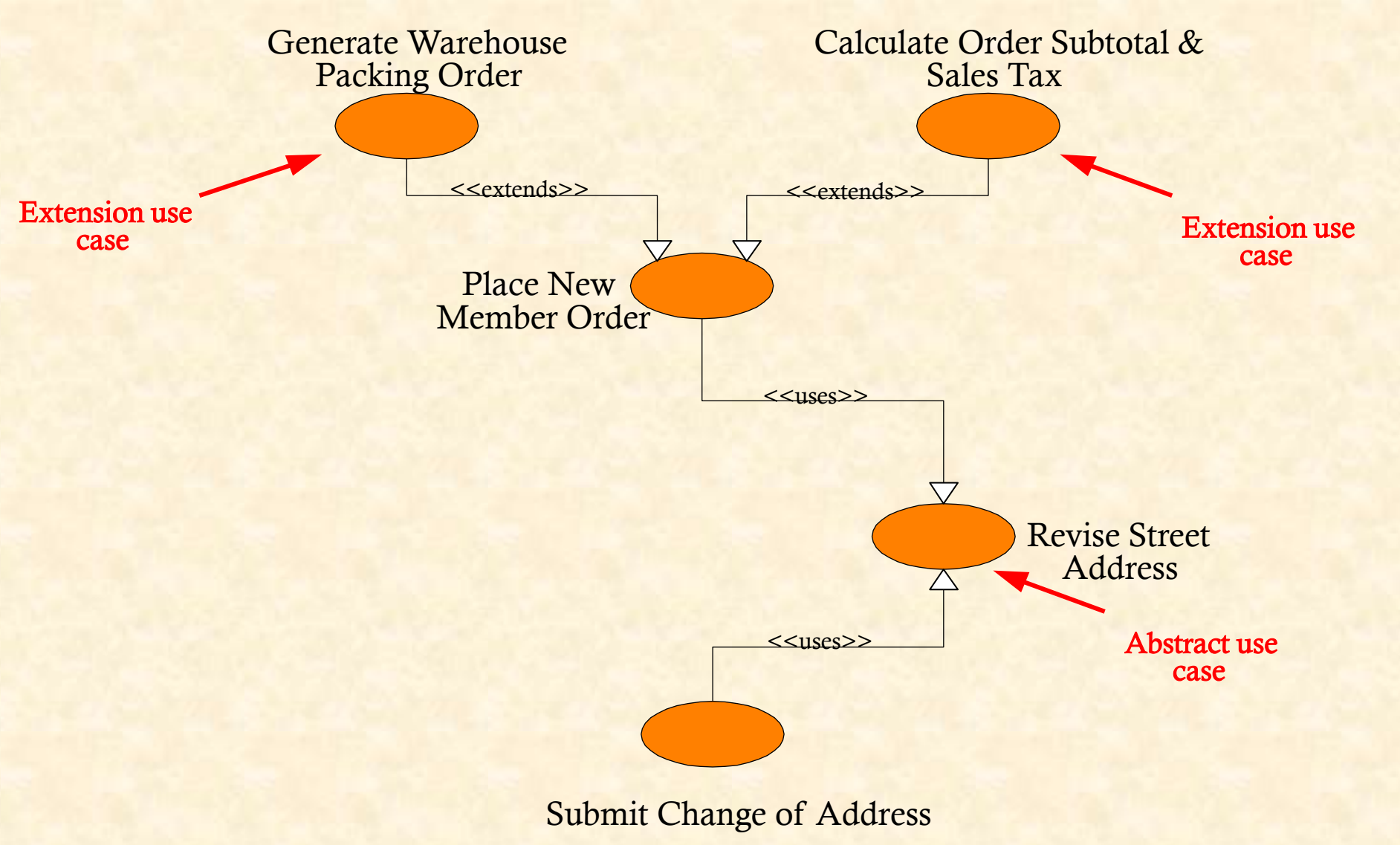
Step 3: Documenting the Use Case Typical Course

Alternate Courses: 3	**Step 2**: If the club member has indicated an address or telephone number change on the promotion order, update the club member's record with the new information. **Step 3**: If Accounts Receivable returns a credit status that the customer is in arrears, send an order rejection notice to the member. **Step 4**: If the product number is not valid, send a notification to the member requesting them to submit a valid product number. If the product being ordered is not available, record the ordered product information and mark as "back-ordered."
Pre-condition: 4	Orders can only be submitted by members.
Post-condition: 5	Member order has been recorded and the picking ticket has been routed to the warehouse.
Assumptions: 6	None at this time.

Extension and Abstract Use Cases

- An extension use case extends the functionality (typical course) of an original use case. An extension use case *can only be invoked by the use case it is extending*.
- An abstract use case contains typical course steps that were common to two or more original use cases. An abstract use case reduces redundancy and promotes reuse.

Depicting Extension and Abstract Use Cases



Step 4: Defining the Analysis Use Case

ANALYSIS USE CASE

Author: S. Shepherd **Date: 10/25/2000**

USE CASE NAME:	Place New Member Order	
ACTOR(S):	Club Member	
DESCRIPTION:	This use case describes the process of a club member submitting a new order for SoundStage products. On completion, the club member will be sent a notification that the order was accepted.	
REFERENCES	MSS-1.0	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1: This use case is initiated when a member submits an order to be processed.	**Step 2**: The member's personal information such as address and phone number is validated against what is currently on file. **Step 3**: For each product being ordered, validate the product number. **Step 4**: For each product being ordered, check the availability in inventory and record the ordered product information such as the quantity being ordered. **Step 5**: Invoke extension use case *Calculate Order Subtotal & Sales Tax*. **Step 6**: The member's credit card information is verified based on the amount due and Accounts Receivable transaction data is checked to make sure no payments are outstanding. **Step 7**: Invoke extension use case *Generate Warehouse Packing Order*. **Step 8**: Generate an order confirmation notice indicating the status of the order and send it to the member.
	Step 9: This use case concludes when the member receives the order confirmation notice.	

(continued)

Step 4: Defining the Analysis Use Case

ALTERNATE COURSES:	**Step 2**: If the club member has indicated an address or telephone number change on the order, invoke abstract use case *Revise Street Address*. **Step 3**: If the product number is not valid, send a notification to the member requesting the member to submit a valid product number. **Step 4**: If the product being ordered is not available, record the ordered product information and mark the order as "backordered." **Step 6**: If member's credit card information is invalid or if member is found to be in arrears, a credit problem notice is sent to the member. Modify the order's status to be "on hold pending payment."
PRECONDITION:	Orders can only be submitted by members.
POSTCONDITION:	Member order has been recorded and the Packing Order has been routed to the Warehouse.
ASSUMPTIONS:	None at this time.

Finding and Identifying Business Objects

- Step 1: Find the Potential Objects
  - Underlining (or highlighting) the use case nouns
- Step 2: Select the Proposed Objects
  - Removing the nouns that represent:
    - Synonyms
    - Nouns outside the scope of the system
    - Nouns that are roles without unique behavior or are external roles
    - Unclear nouns that need focus
    - Nouns that are really actions or attributes

Use Case with Nouns Highlighted

Author: S. Shepherd | **ANALYSIS USE CASE** | **Date: 10/25/2000**

USE CASE NAME:	Place New Member Order	
ACTOR(S):	Club Member	
DESCRIPTION:	This use case describes the process of a club member submitting a new order for SoundStage products. On completion, the club member will be sent a notification that the order was accepted.	
REFERENCES	MSS-1.0	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	Step 1: This use case is initiated when a member submits an order to be processed.	**Step 2**: The member's personal information such as address and phone number is validated against what is currently on file.
		Step 3: For each product being ordered, validate the product number.
		Step 4: For each product being ordered, check the availability in inventory and record the ordered product information such as the quantity being ordered.
		Step 5: Invoke extension use case *Calculate Order Subtotal & Sales Tax*.
		Step 6: The member's credit card information is verified based on the amount due and Accounts Receivable transaction data is checked to make sure no payments are outstanding.
		Step 7: Invoke extension use case *Generate Warehouse Packing Order*.
	Step 9: This use case concludes when the member receives the order confirmation notice.	**Step 8**: Generate an order confirmation notice indicating the status of the order and send it to the member.

(continued)

Use Case with Nouns Highlighted

ALTERNATE COURSES:	**Step 2**: If the club member has indicated an address or telephone number change on the order, invoke abstract use case *Revise Street Address*. **Step 3**: If the product number is not valid, send a notification to the member requesting the member to submit a valid product number. **Step 4**: If the product being ordered is not available, record the ordered product information and mark the order as "backordered." **Step 6**: If member's credit card information is invalid or if member is found to be in arrears, a credit problem notice is sent to the member. Modify the order's status to be "on hold pending payment."
PRECONDITION:	Orders can only be submitted by members.
POSTCONDITION:	Member order has been recorded and the Packing Order has been routed to the Warehouse.
ASSUMPTIONS:	None at this time.

Potential Objects Extracted from Use Case

POTENTIAL OBJECT LIST
Accounts Receivable Department
Amount Due
Club Member
Credit Card Information
Credit Problem Notice
Credit Status
File
Marketing Department
Member Address
Member Order
Member Phone Number
Member Services Department
Member Services System
Order
Order Confirmation Notice
Order Sales Tax
Order Status
Order Subtotal
Ordered Product
Ordered Product Information
Ordered Product Quantity
Past Member
Payments
Potential Member
Product
Product Inventory
Product Number
Street Address
Transaction
Warehouse
Warehouse Packing Order

Analysis of the Potential Objects

POTENTIAL OBJECT LIST		REASON
Accounts Receivable Department	x	Not relevant for current project
Amount Due	x	Attribute of "MEMBER ORDER"
Club Member	✓	Type of "MEMBER"
Credit Card Information	x	Attribute of "MEMBER"
Credit Problem Notice	x	Potential Interface item to be addressed in object-oriented design
Credit Status	x	Attribute of "MEMBER"
File	x	Not relevant for current project
Marketing Department	x	Not relevant for current project
Member Address	x	Attribute of "MEMBER"
Member Order	✓	"MEMBER ORDER"
Member Phone Number	x	Attribute of "MEMBER"
Member Services Department	x	Not relevant for current project
Member Services System	x	Not relevant for current project
Order	x	Another name for "MEMBER ORDER"
Order Confirmation Notice	x	Potential Interface item to be addressed in object-oriented design
Order Sales Tax	x	Attribute of "MEMBER ORDER"
Order Status	x	Attribute of "MEMBER ORDER"
Order Subtotal	x	Attribute of "MEMBER ORDER"
Ordered Product	✓	"MEMBER ORDERED PRODUCT"
Ordered Product Information	x	Unclear noun
Ordered Product Quantity	x	Attribute of "MEMBER ORDERED PRODUCT"
Past Member	✓	Type of "MEMBER"
Payments	✓	Type of "TRANSACTION"
Potential Member	✓	Type of "MEMBER"
Product	✓	"PRODUCT"
Product Inventory	x	Attribute of "PRODUCT"
Product Number	x	Attribute of "PRODUCT"
Street Address	x	Attribute of "MEMBER"
Transaction	✓	"TRANSACTION"
Warehouse	x	Not relevant for current project
Warehouse Packing Order	x	Potential Interface item to be addressed in object-oriented design

Results of Analysis

PROPOSED OBJECT LIST
CLUB MEMBER
MEMBER ORDER
MEMBER ORDERED PRODUCT
PAST MEMBER
PAYMENT
POTENTIAL MEMBER
PRODUCT
TRANSACTION
-- PLUS --
AGREEMENT
AUDIO TITLE
GAME TITLE
PROMOTION
MERCHANDISE
RETURN
TITLE
VIDEO TITLE

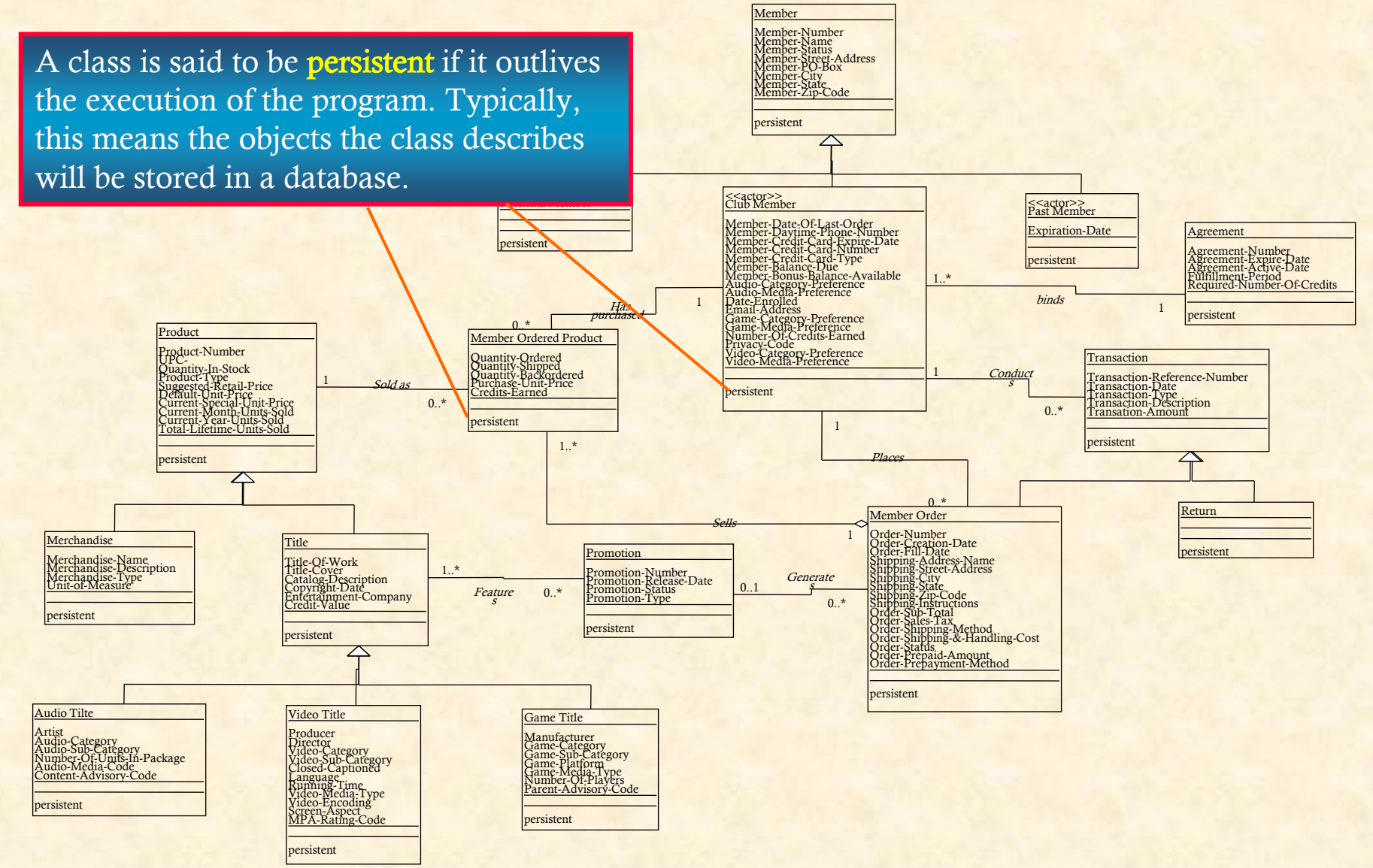
Having found in other use cases

Constructing a Class Diagram

- Step 1: Identify Associations and Multiplicity
  - Use an object/class matrix
- Step 2: Identify Generalization/Specialization Relationships
- Step 3: Identify Aggregation Relationships
- Step 4: Prepare the Class Diagram

Member Services System Class Diagram

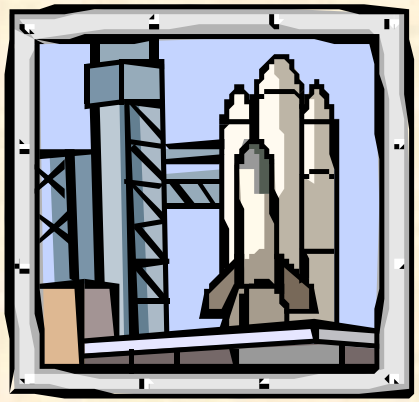
A class is said to be **persistent** if it outlives the execution of the program. Typically, this means the objects the class describes will be stored in a database.



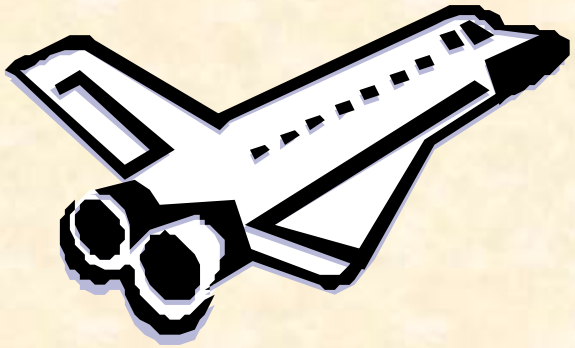
Modeling the Behavior of the Objects

Object State Example

Possible "States" of the Space Shuttle



"Takeoff"



"PRE-LAUNCH" state

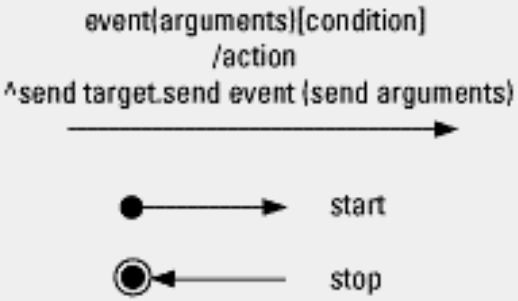
"FLIGHT" state

State Diagram

STATE-TRANSITION DIAGRAM Shows the state space of a given context, the events that cause a transition from one state to another, and the actions that result

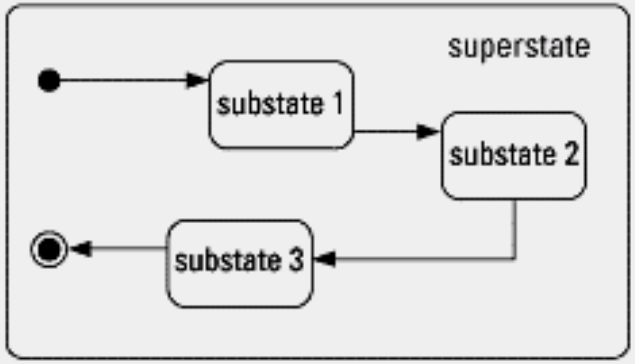
State icon

State transitions

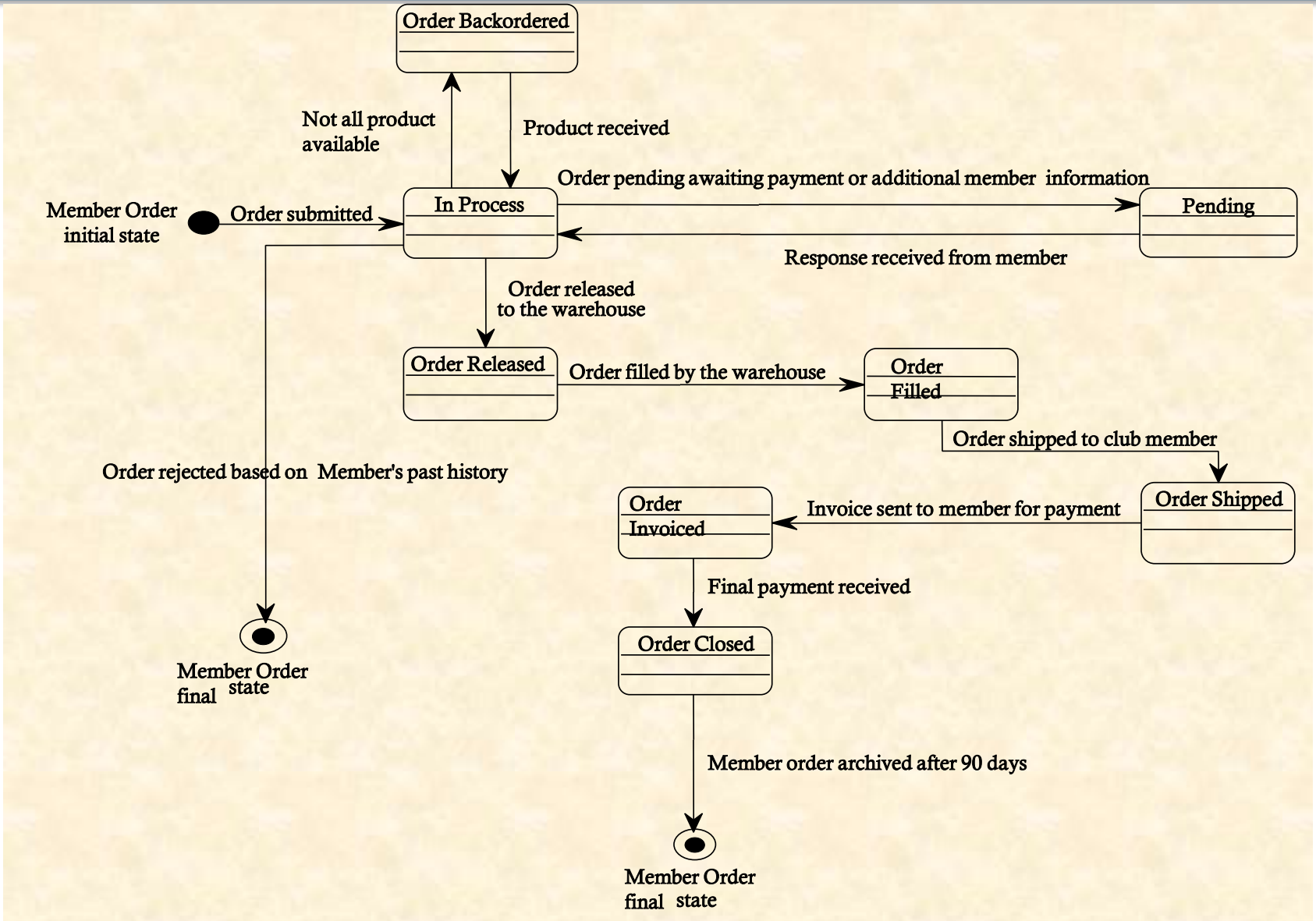


History Ⓗ

Nesting



Member Order State Diagram



要点与引申

- UML本身是一种工具，渗透了面向对象的基本思想。将UML表示应用到具体活动的具体建模过程中，可以从已有的范例（特别是分析模式和设计模式）中汲取经验。
- 利用 Use case 进行功能性需求建模，首先要完整地表达出用户需求，其次还要帮助用户对需求进行归纳、抽象和整理，后者体现在对 Use case 的适度分解与相互关联定义上。用例建模是从本质功能到功能细节渐进的发现和认识过程。
- 识别 Actor 和 Use cases、阅读和书写高层次的和详细格式的 Use cases、区分本质的 Use cases 和 现实的 Use cases，这些技能是利用 Use case 进行功能性需求建模的关键。