西安电子科技大学
XIDIAN UNIVERSITY

B

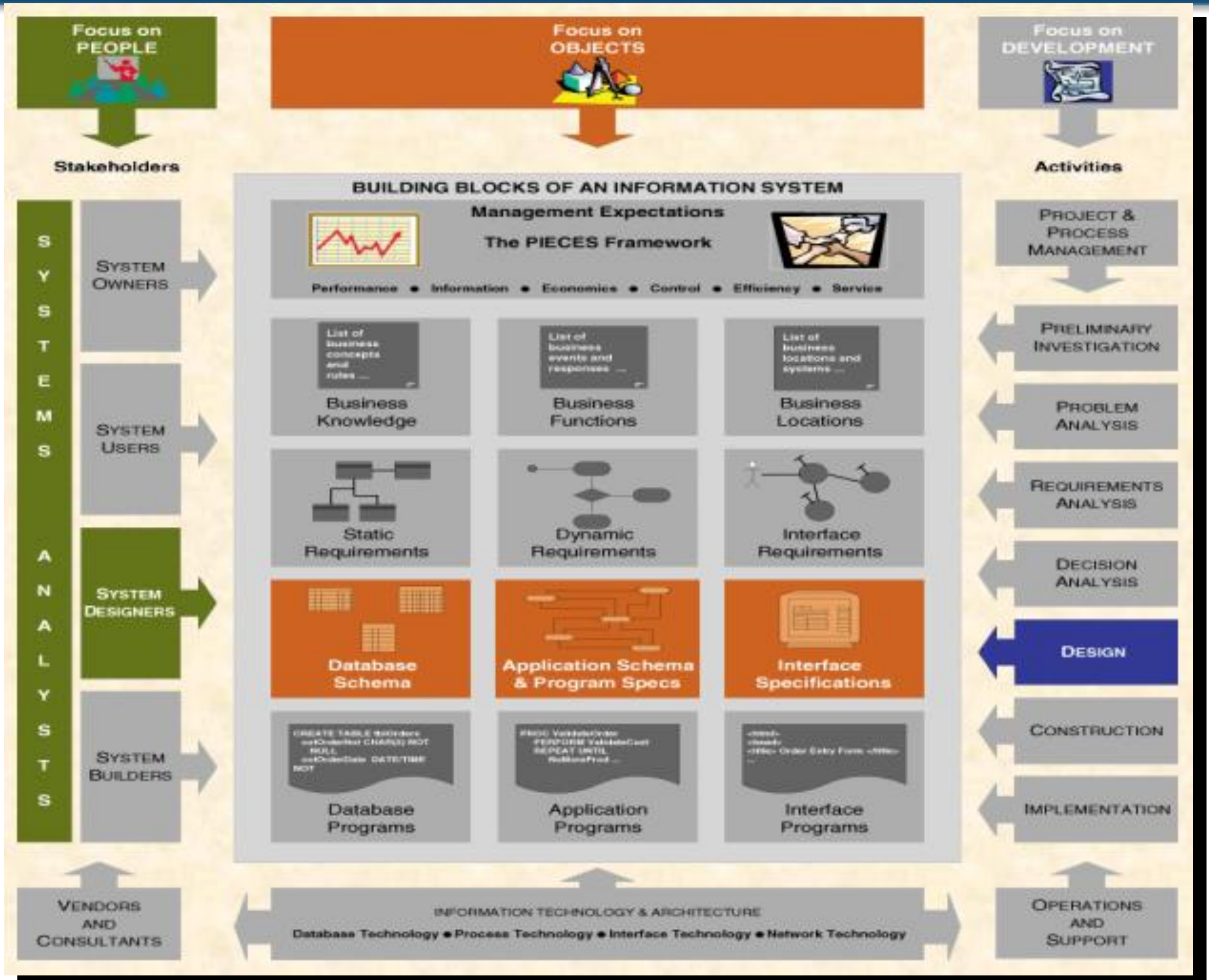
系统分析与设计
（System Analysis and Design）

Object-oriented Design and Modeling

Content Structure

- An Introduction to Object-Oriented Design
- The Process of Object-Oriented Design

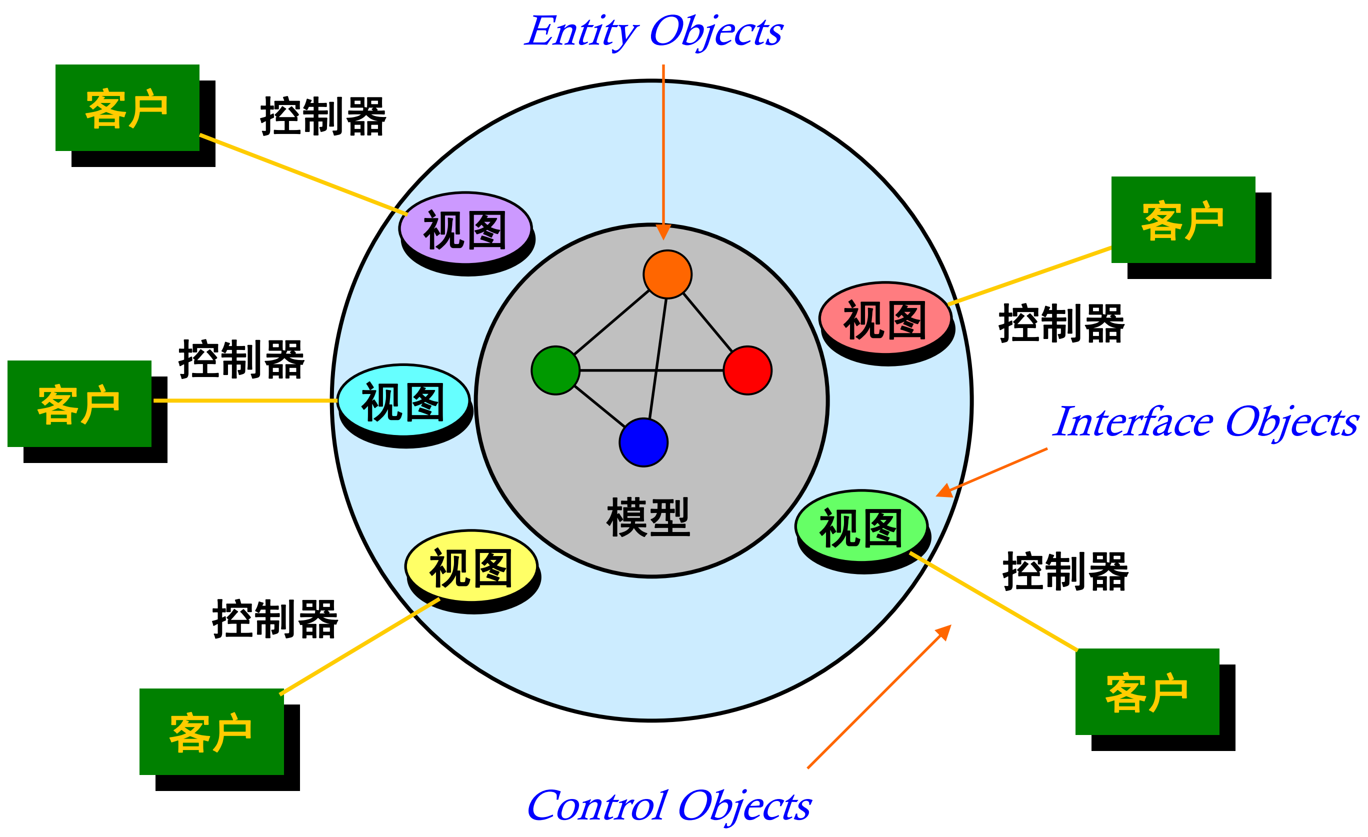
Chapter Map


Focus on PEOPLE
Focus on OBJECTS
Focus on DEVELOPMENT
Stakeholders
Activities
SYSTEMS ANALYSTS
SYSTEM OWNERS
SYSTEM USERS
SYSTEM DESIGNERS
SYSTEM BUILDERS
BUILDING BLOCKS OF AN INFORMATION SYSTEM
Management Expectations
The PIECES Framework
Performance • Information • Economics • Control • Efficiency • Service
Business Knowledge
Business Functions
Business Locations
Static Requirements
Dynamic Requirements
Interface Requirements
Database Schema
Application Schema & Program 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-Oriented Design

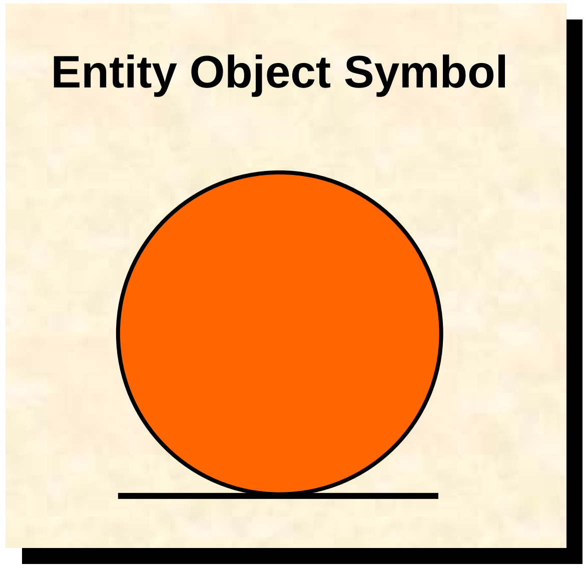
MVC Pattern

- Smalltalk 的 MVC（Model-Views-Controller）结构对以后软件环境的结构设计乃至软件设计产生了深刻的影响。



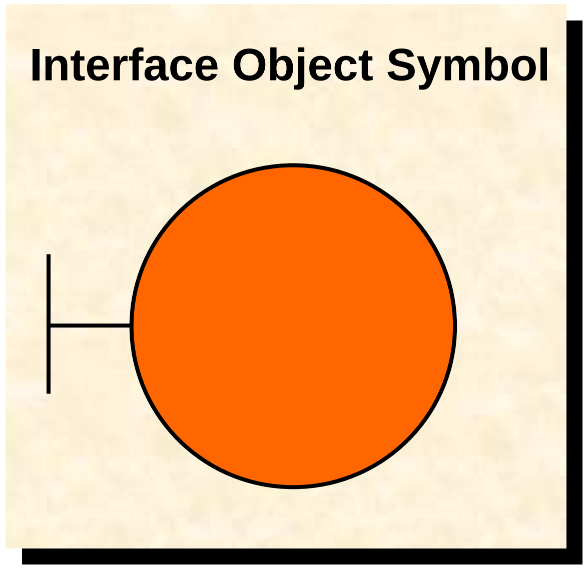
Design Object Types - Entity Objects

- The objects that *represent actual data* within the business domain in which the user is interested in *storing* are called entity objects.
  - Be found during analysis and refined during design.



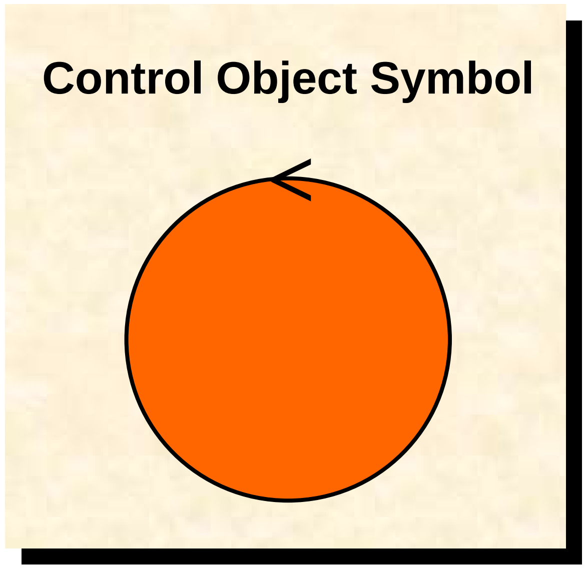
Design Object Types - Interface Objects

- Objects that *represent a means* through which the user will interface with the system are called interface objects.
  - Be introduced during design.



Design Object Types - Control Objects

- Objects that hold application or business rule logic are called control objects.
  - Be introduced during design.



Object Responsibilities

- An object responsibility is the obligation（义务） that an object has to provide a service when requested and thus collaborating with other objects to satisfy the request if required.

Object Responsibilities

- Responsibilities of different types of objects:
  - Interface objects
    - *translate* the user's input into information that the system can understand and use to process the business event.
    - *take* data pertaining to（关于） a business event and *translates* the data for appropriate presentation to the user.
  - Entity objects
    - *correspond to* items in real life*, contain* information, known as attributes, that describes the different instances of the entity, and *store* it persistently.
  - Control objects
    - serve as the "traffic cop"（交通警） containing the application logic or business rules of the event for *managing or directing* the interaction between objects.

FileReader *文件读取类*

Member *会员信息类*

Request *消息请求类*

Object Reusability

- The number one driving force for developing systems using object-oriented technology is *the ability to reuse objects*. （使用面向对象技术开发系统的第一推动力是复用对象的能力。）
- Results of a study performed by EDS using two different technologies (one traditional, one OO) to develop the same system.

PL/1	19 calendar months	152 person months	265,000 LOC
Smalltalk	3.5 calendar months	10.4 person months	22,000 LOC

**LOC = Lines of code*

Object Reusability

- An object framework（框架） is a set of related, interacting objects that provide a well-defined set of services for accomplishing a task.
  - Example:
    - Microsoft MFC
- A component（构件/组件） is a group of objects packaged together into one unit (with a standard interface).
  - Examples:
    - A dynamic link library (DLL) or executable file
    - Microsoft ActiveX
    - Java Beans

The Process of Object-Oriented Design

Object-Oriented Design Activities

- Refining the use case model to reflect the implementation environment.
- Modeling object interactions and behavior that support the use case scenario.
- Updating the object model to reflect the implementation environment.

Refining The Use Case Model

- Step 1: Transforming the “Analysis” Use Cases to “Design” Use Cases
- Step 2: Updating the Use Case Model Diagram and Other Documentation

Module A - Object-Oriented Analysis and Modeling

ANALYSIS USE CASE

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

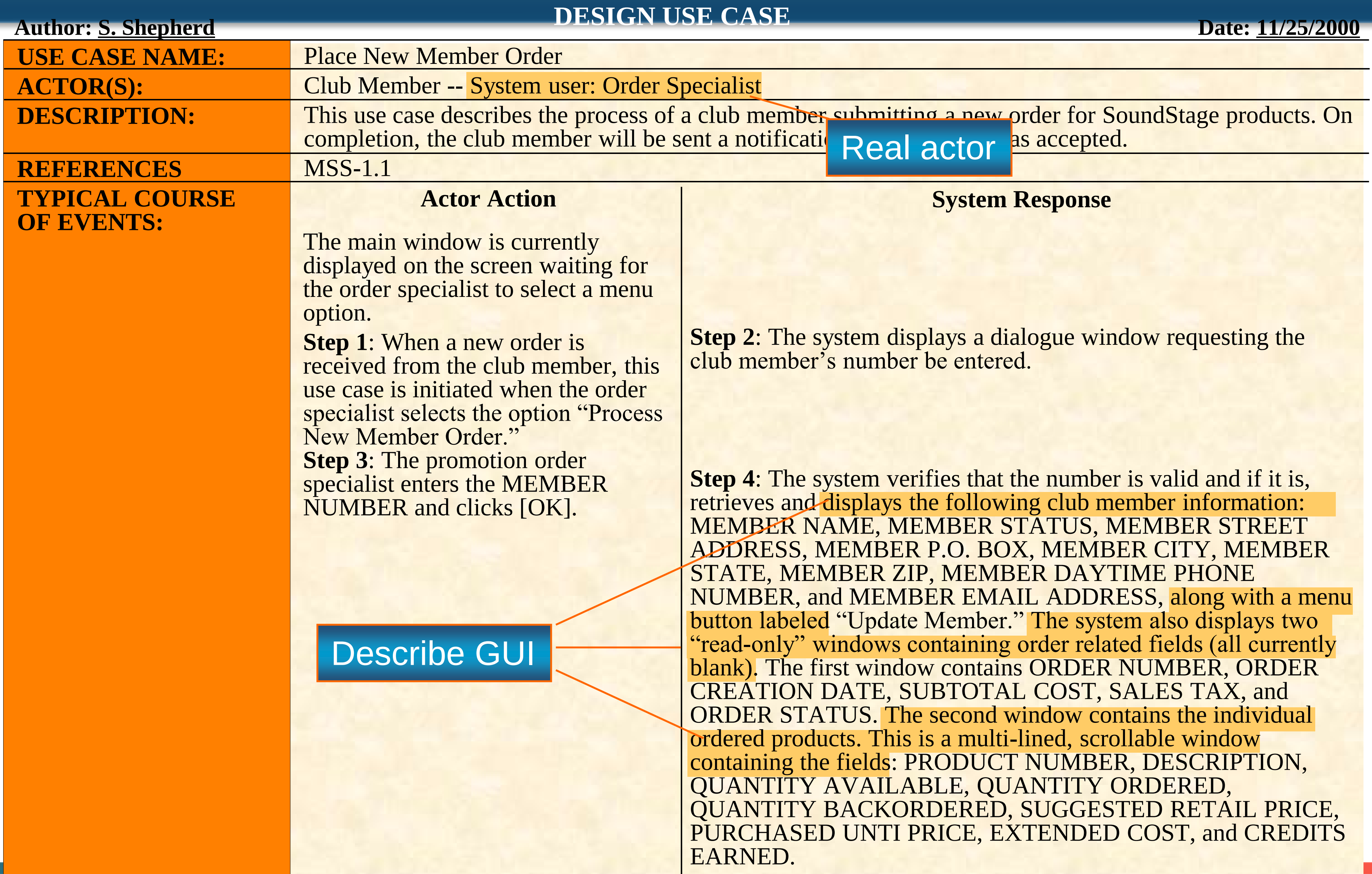
USE CASE NAME:	Place New Member Order	
ACTOR(S):	Club Member	
DESCRIPTION:	This use case describes the process of a club member submitting a new order for SoundStage products. On completion, the club member will be sent a notification that the order was accepted.	
REFERENCES	MSS-1.0	
TYPICAL COURSE OF 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 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)

Module A - Object-Oriented Analysis and Modeling

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.

Step 1: Defining the Design Use Case



DESIGN USE CASE

Author: S. Shepherd **Date:** 11/25/2000

USE CASE NAME:	Place New Member Order	
ACTOR(S):	Club Member -- System user: Order Specialist	
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 [the order w]as accepted.	
REFERENCES	MSS-1.1	
TYPICAL COURSE OF EVENTS:	**Actor Action**	**System Response**
	The main window is currently displayed on the screen waiting for the order specialist to select a menu option.	
	Step 1: When a new order is received from the club member, this use case is initiated when the order specialist selects the option "Process New Member Order."	**Step 2**: The system displays a dialogue window requesting the club member's number be entered.
	Step 3: The promotion order specialist enters the MEMBER NUMBER and clicks [OK].	**Step 4**: The system verifies that the number is valid and if it is, retrieves and displays the following club member information: MEMBER NAME, MEMBER STATUS, MEMBER STREET ADDRESS, MEMBER P.O. BOX, MEMBER CITY, MEMBER STATE, MEMBER ZIP, MEMBER DAYTIME PHONE NUMBER, and MEMBER EMAIL ADDRESS, along with a menu button labeled "Update Member." The system also displays two "read-only" windows containing order related fields (all currently blank). The first window contains ORDER NUMBER, ORDER CREATION DATE, SUBTOTAL COST, SALES TAX, and ORDER STATUS. The second window contains the individual ordered products. This is a multi-lined, scrollable window containing the fields: PRODUCT NUMBER, DESCRIPTION, QUANTITY AVAILABLE, QUANTITY ORDERED, QUANTITY BACKORDERED, SUGGESTED RETAIL PRICE, PURCHASED UNTI PRICE, EXTENDED COST, and CREDITS EARNED.

Real actor

Describe GUI

Step 1: Defining the Design Use Case (continued)

TYPICAL COURSE OF EVENTS:	**Step 5**: The order specialist checks to see if the club member has made any address or phone number changes on the order. If changes have been made the order specialist clicks the [Update member] menu button - invoke abstract use case *Revise Street Address*. Otherwise, the order specialist clicks [Enter Ordered Product] menu button.	**Step 6**: The System displays a dialogue window prompting the user to enter the PRODUCT NUMBER.
	Step 7: The order specialist enters the PRODUCT NUMBER of the item being ordered and then presses [Enter].	**Step 8**: The system validates the product number and then retrieves and displays the DESCRIPTION, CREDIT VALUE, QUANTIY IN STOCK, and SUGGESTED RETAIL PRICE. The cursor is then advanced to the QUANTITY ORDERED field.
	Step 9: The order specialist enters the QUANTITY ORDERED and a PURCHASE UNIT PRICE if it differs from the SUGGESTED RETAIL PRICE and presses [Enter]. If PURCHASE UNIT PRICE is not entered it will default to the SUGGESTED RETAIL PRICE.	**Step 10**: The system validates the inputs then calculates the new EXTENDED COST by multiplying the PURCHASE UNIT PRICE times the QUANTITY ORDERED. If the QUANTITY ORDERED is greater than the QUANTITY IN STOCK, the system calculates QUANTITY BACKORDERED by subtracting QUANTITY IN STOCK from QUANTITY ORDERED. The system displays all fields in the ordered product read-only window.
		Step 11: The system displays a window with the message "Is the Ordered Product Information Correct?" If no go to **Step 7**. If yes the system displays another window with the message "Additional Products to be Entered?"
	Step 12: If the order specialist click [Yes] go back to **Step 5**.	**Step 13**: The system assigns a unique ORDER NUMBER, and an ORDER STATUS of "O," then calculates ORDER SUBTOTAL and ORDER SALES TAX - invoke extension use case *Calculate Order Subtotal & Sales Tax*. The system displays all fields in the Order read-only window and then displays the message "Commit Order?" - [Yes] or [Cancel].

Related use cases

Describe GUI

Step 1: Defining the Design Use Case (continued)

TYPICAL COURSE OF EVENTS:	**Step 14**: The order specialist clicks [Yes]. **Step 17**: This use case concludes when the system displays a message "Order 1237645 has been processed successfully, would you like to enter another?" If the order specialist clicks [Yes] the use case is repeated. If the order specialist clicks [No], the main system window is displayed.	**Step 15**: The system reduces the QUANTITY IN STOCK by the QUANTITY ORDERED for each product being ordered (replace any negative result with zero) and then saves all entries. Then the system verifies the member's status and creditworthiness - invoke abstract use case *Verify Club member Status*. **Step 16**: The system generates a packing order - invoke extension use case *Generate Warehouse Packing Order*, and generates an order confirmation notice indicating the status of the order - invoke abstract use case *Generate Order Confirmation Notice*. If there were and invalid entries submitted by the club member invoke abstract use case *Generate Order Error Report*. Related use cases Error messages
ALTERNATE COURSES:	**Step 4**: If the MEMBER NUMBER is invalid, the system displays a window with the error message "Member Number Not on File." The order specialist can then re-enter the number or cancel the transaction. **Step 8**: If the PRODUCT NUMBER is not valid, the system displays a window with the error message "Product Number not Valid." The order specialist either corrects the entry or advances to the next entry. In a later step, an error report will be generated for any invalid entries. **Step 10**: If the QUANTITY ORDERED or PURCHASE UNIT PRICE is not valid, the system highlights the field(s) in error and then displays a window with the error message "Highlighted fields invalid." The order specialist either corrects the entry or advances to the next entry. In a later step an error report will be generated for any invalid entries. **Step 13**: If any of the products being ordered is not available, record the order status as "P" (partially filled). **Step 16**: If member's credit card information is invalid or if the member is found to be in arrears, a credit problem notice is sent to the member. Modify the order's status to be "H" - on hold pending payment. Exit transaction.	
PRECONDITION:	Order specialist has logged on the system and has authorization for this transaction.	
POSTCONDITION:	Member order has been recorded, the Packing Order has been routed to the Warehouse, and a confirmation notice has been generated and sent to the club member.	
ASSUMPTIONS:	None at this time.	

Step 2: Updating the Use Case Model Diagram

- After all the analysis use cases have been transformed to design use cases, it is possible that *new use cases or even actors have been discovered*. It is very important that we keep our documentation accurate and current. Thus, in this step the use case model diagram and the actor and use case glossaries should be updated to reflect any new information introduced in step 1.

Modeling Object Interactions and Behaviors

- Step 1: Identify and Classify Use Case Design Objects
- Step 2: Identify Object Attributes
- Step 3: Model High-Level Object Interactions
- Step 4: Identify Object Behaviors and Responsibilities
- Step 5: Model Detailed Object Interactions

Step 1: Classify Design Objects

INTERFACE OBJECTS	CONTROLLER OBJECTS	ENTITY OBJECTS
Member Services Main Window	Order Processor	CLUB MEMBER
Club Member Number Dialogue Window	Packing Order Generator	MEMBER ORDER
Order Processing Main Window		MEMBER ORDERED PRODUCT
Club Member Window		PRODUCT
Order Window		
Member Ordered Products Window		
Product Dialogue Window		
Message Window		
Warehouse Printer		

Step 2: Identify Object Attributes

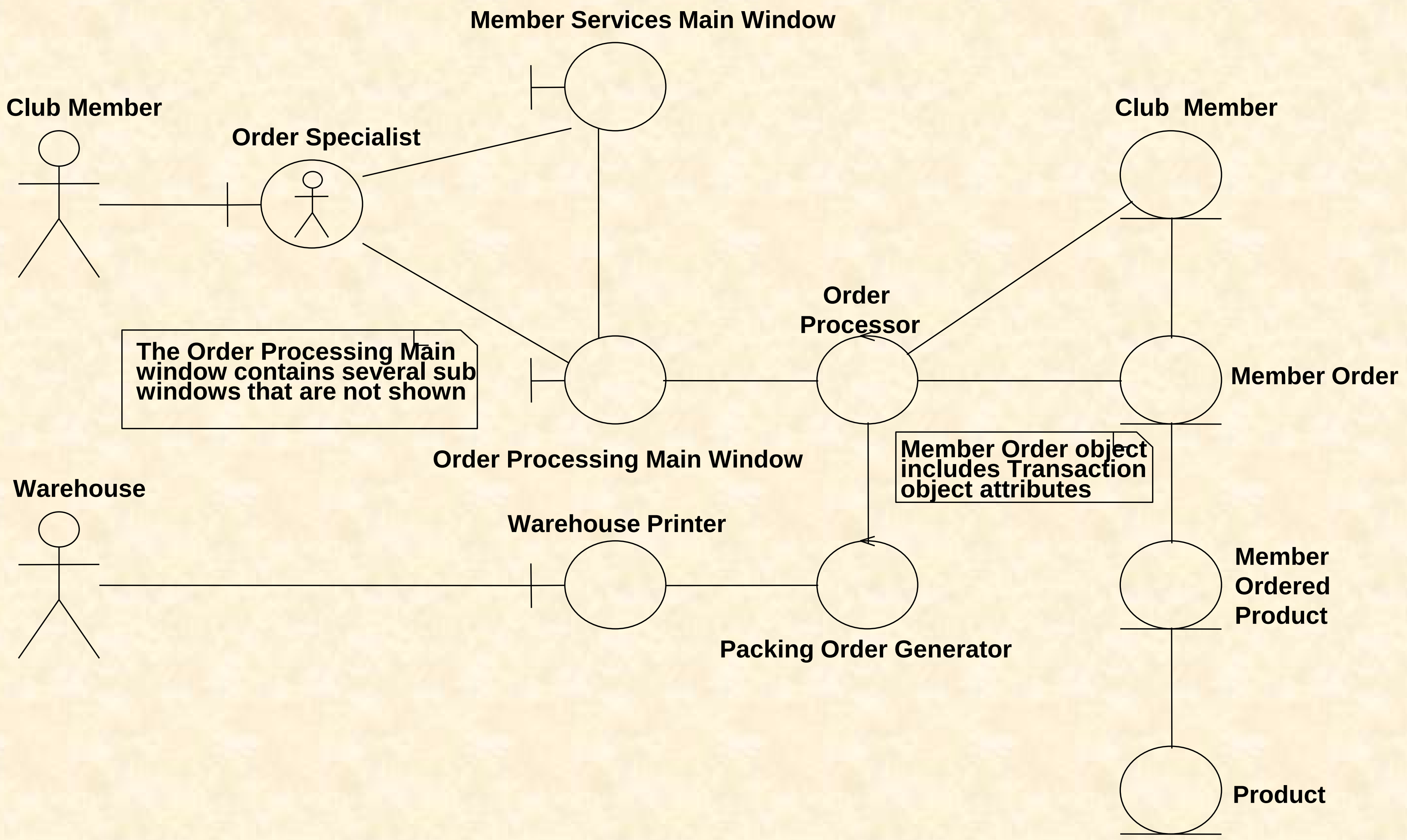
- Reference the attributes in the use case text.
- Examine each use case for additional attributes that haven’t been previously identified.
- Update the object class diagram to include those attributes.

Step 3: Model High-Level Interactions

- After identifying and categorizing the design objects involved in a use case, we need to model those objects and their interactions. Such models are called ideal object model diagrams.
- An ideal object model diagram includes symbols to represent actors, interface, control and entity objects, and lines that represent messages or communication between the objects.

Step 3: Model High-Level Interactions

Construct Ideal Object Model Diagram


Member Services Main Window
Club Member
Order Specialist
Club Member
Order Processor
Member Order
The Order Processing Main window contains several sub windows that are not shown
Order Processing Main Window
Member Order object includes Transaction object attributes
Warehouse
Warehouse Printer
Member Ordered Product
Packing Order Generator
Product

Step 4: Identify Object Behaviors and Responsibilities

- Task 1: Analyze the use cases to identify required system behaviors.
- Task 2: Associate behaviors and responsibilities with objects.
- Task 3: Examine object model for additional behaviors.
- Task 4: Verify classifications.

Task 1: Identify Behaviors and Responsibilities

BEHAVIORS	AUTOMATED/MANUAL	OBJECT TYPE
Process new member order	Manual/Automated	Control
Submit new order	Manual	
Select process new member order option	Manual	
Display club member number dialogue window	Automated	Interface
Enter member number	Manual	
Verify member number	Automated	Entity
Retrieve club member information	Automated	Entity
Display club member information	Automated	Interface
Display blank order window	Automated	Interface
Display blank member ordered products window	Automated	Interface
Check for address changes	Manual	
Click button	Manual	
Display product number dialogue window	Automated	Interface
Enter product number & press enter	Manual	
Validate product number	Automated	Entity
Retrieve product information	Automated	Interface
Advance cursor	Automated	Interface
Enter order information	Manual	
Validate input fields	Automated	Entity
Calculate extended cost	Automated	Entity
Display ordered product information	Automated	Interface
Display message windows (various)	Automated	Interface
Assign unique order number	Automated	Control
Update order status	Automated	Entity
Calculate order subtotal & sales tax	Automated	Entity
Reduce quantity in stock	Automated	Entity
Calculate quantity backordered	Automated	Entity
Verify member status and credit worthiness	Automated	Entity
Generate warehouse packing order	Automated	Control
Generate order confirmation notice	Automated	Control
Generate order error report	Automated	Control

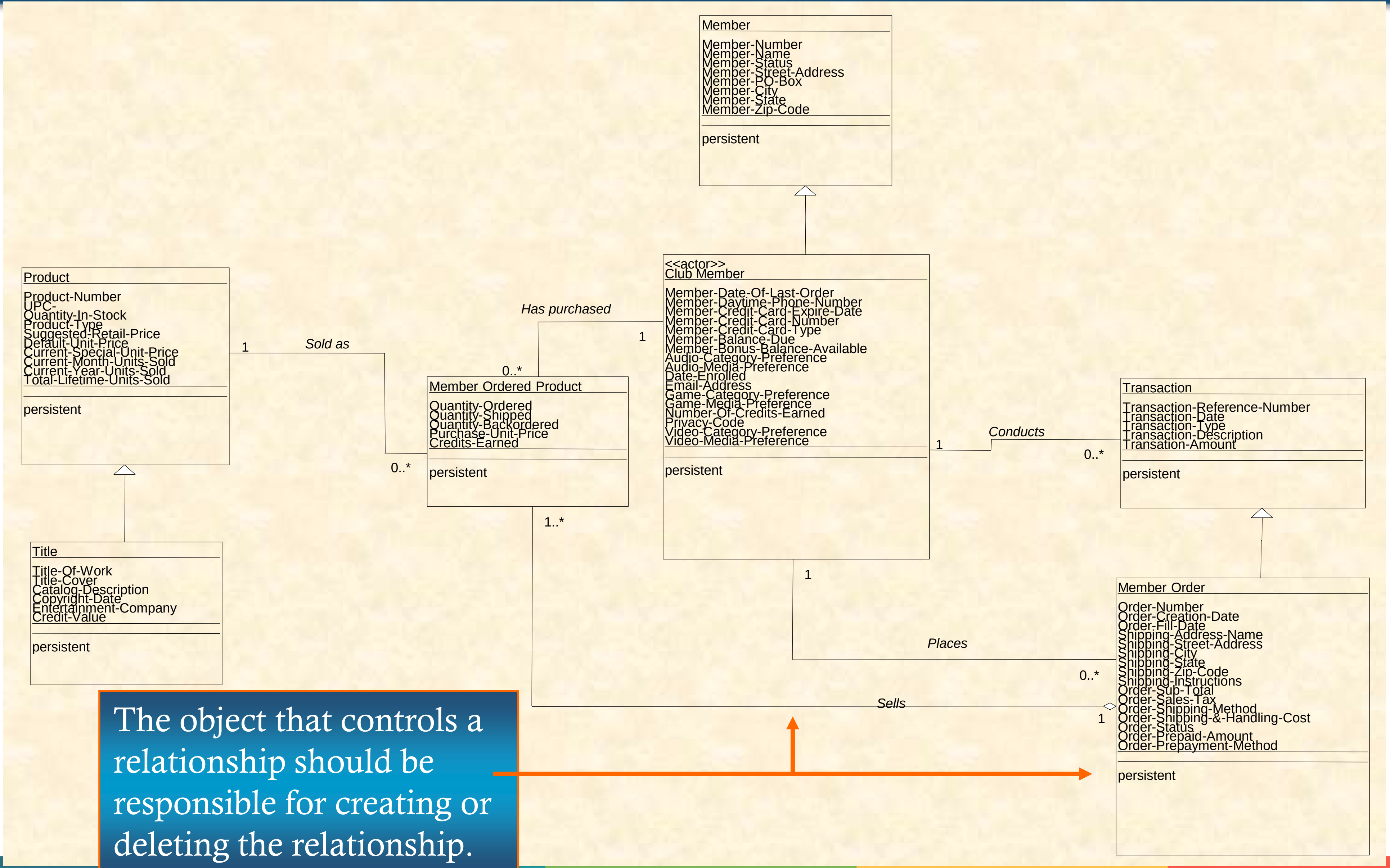
Task 2: Associate Behaviors and Responsibilities

BEHAVIORS	OBJECT TYPE
Process new member order	Control
Assign unique order number	Control
Generate warehouse packing order	Control
Generate order confirmation notice	Control
Generate order error report	Control
Verify member number	Entity
Retrieve club member information	Entity
Validate product number	Entity
Validate input fields	Entity
Calculate extended cost	Entity
Update order status	Entity
Calculate order subtotal & sales tax	Entity
Reduce quantity in stock	Entity
Calculate quantity backordered	Entity
Verify member status and credit worthiness	Entity
Display club member number dialogue window	Interface
Display club member information	Interface
Display blank order window	Interface
Display blank member ordered products window	Interface
Display product number dialogue window	Interface
Retrieve product information	Interface
Advance cursor	Interface
Display ordered product information	Interface
Display message windows (various)	Interface

Example CRC (Class Responsibility Collaboration) Card

Object Name: Member Order	
Sub Object:	
Super Object:	
Behaviors and Responsibilities	**Collaborators**
Report order information	**Member Ordered Product**
Calculate subtotal cost	
Calculate sales tax	
Update order status	
Create ordered product	
Delete ordered product	

Task 3: Examine Object Model for Additional Behaviors



The object that controls a relationship should be responsible for creating or deleting the relationship.

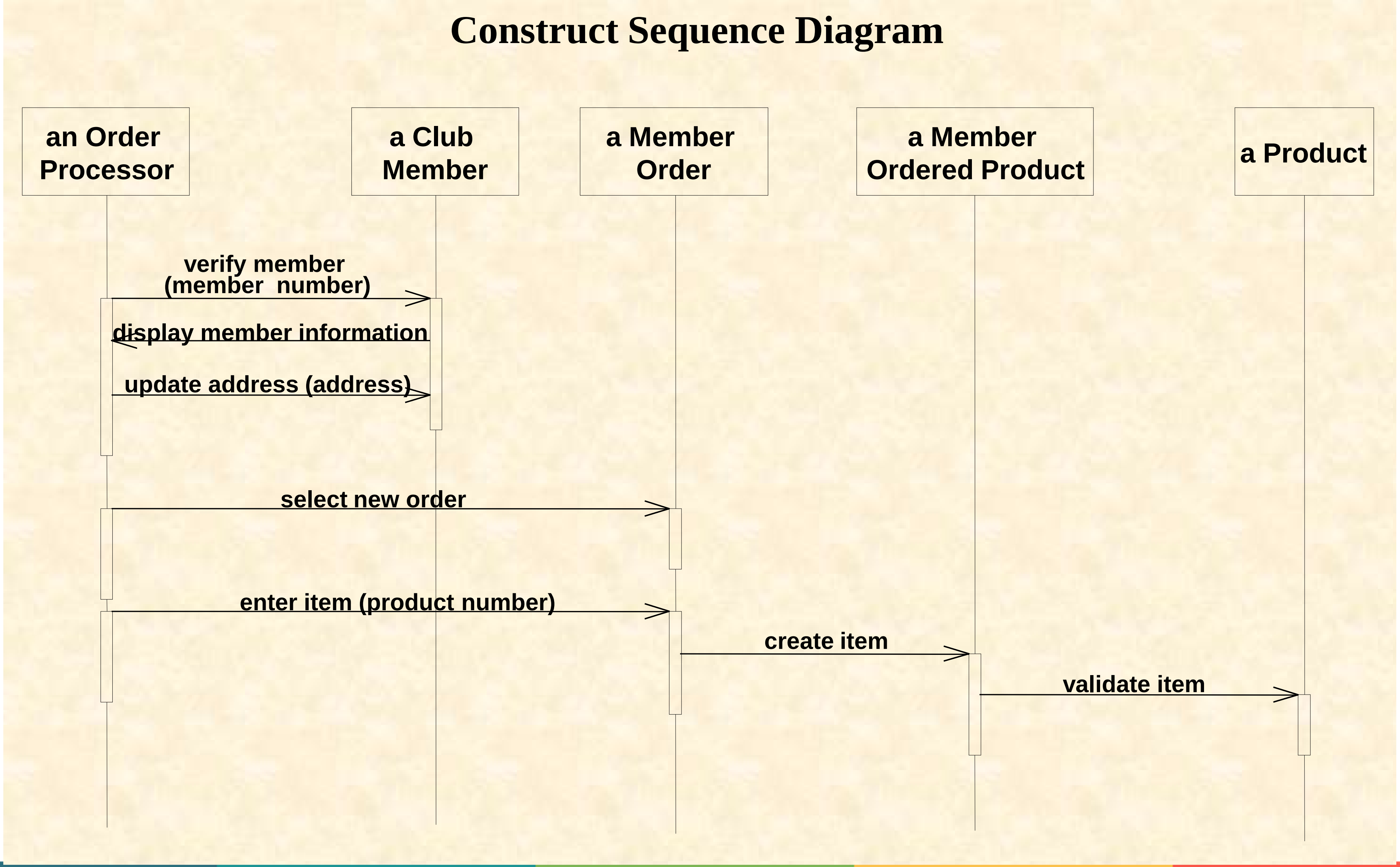
Four Implicit Object Behaviors

- Create new object instances.
- Update object data or attributes.
- Delete object instances.
- Display information (attribute values) about the object.

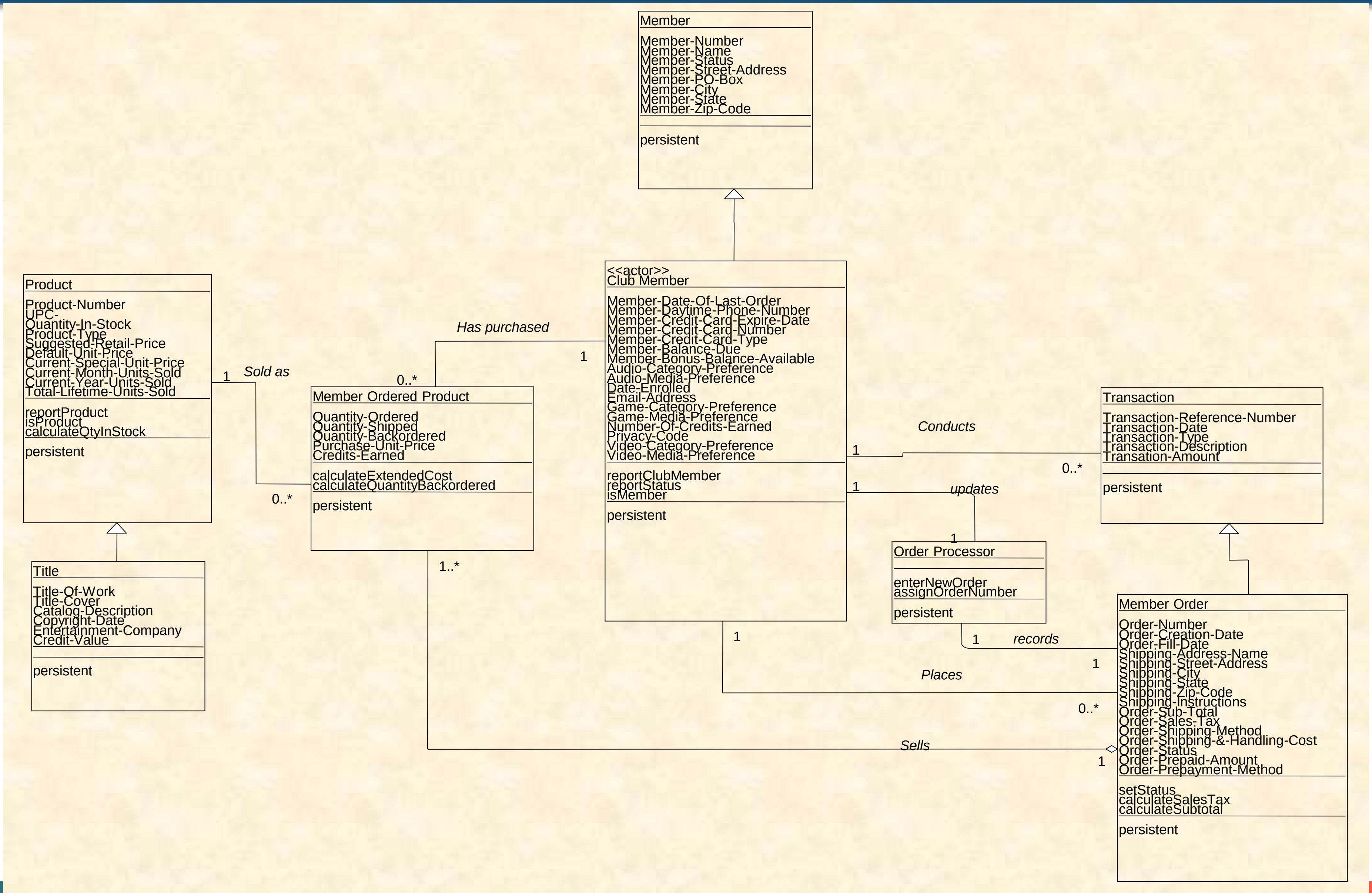
Task 4: Verify Classifications

- A commonly used approach is role playing.
  - The use case scenarios are acted out by the participants.
  - The participants assume the role of actors or object types that collaborate to process a hypothetical business event.
  - Message sending is simulated by using an item such as a ball that is passed between the participants.

Step 5: Model Detailed Interactions



Updating the Object Class Diagram to Reflect Design

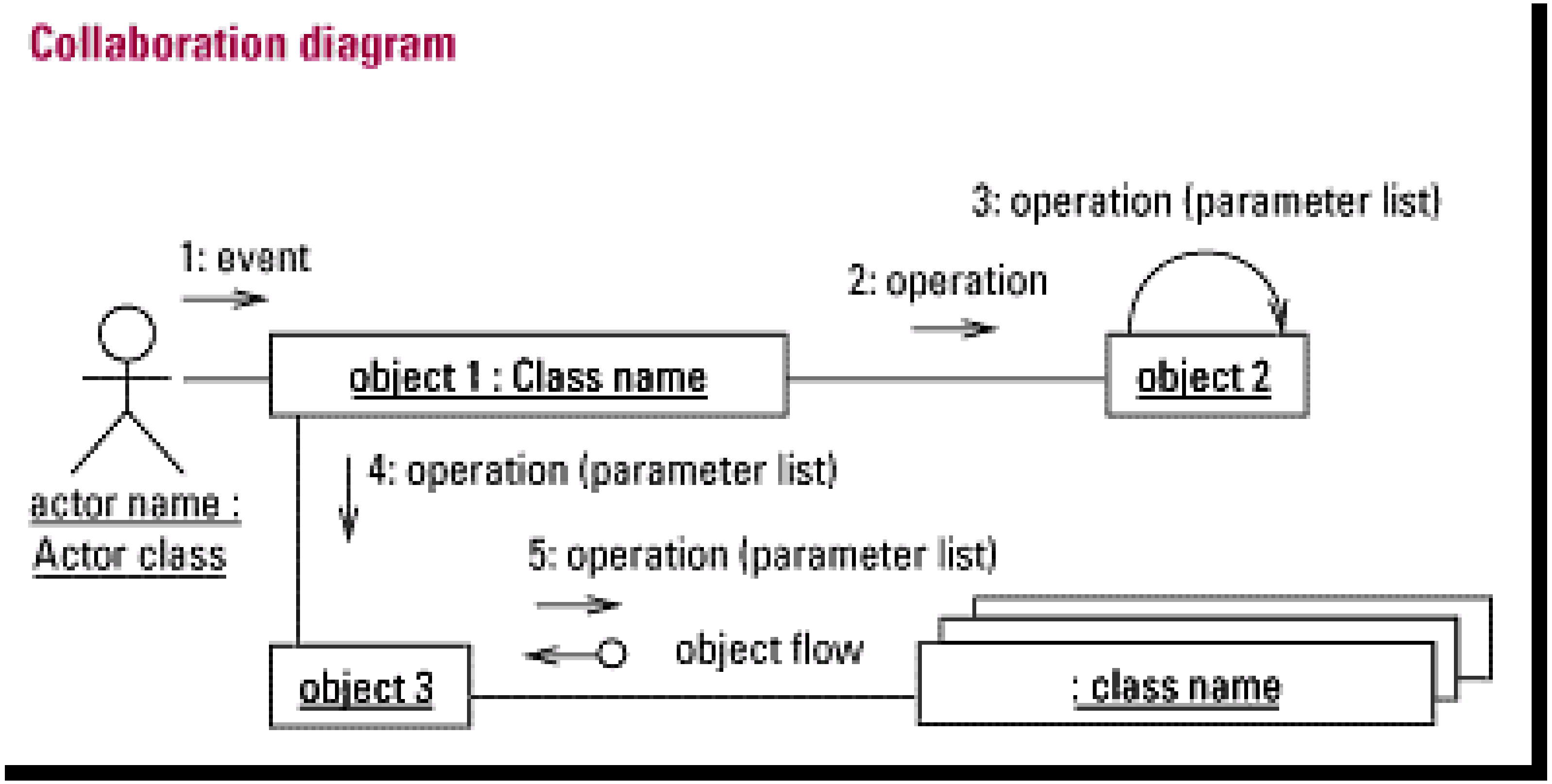


Other UML Diagrams

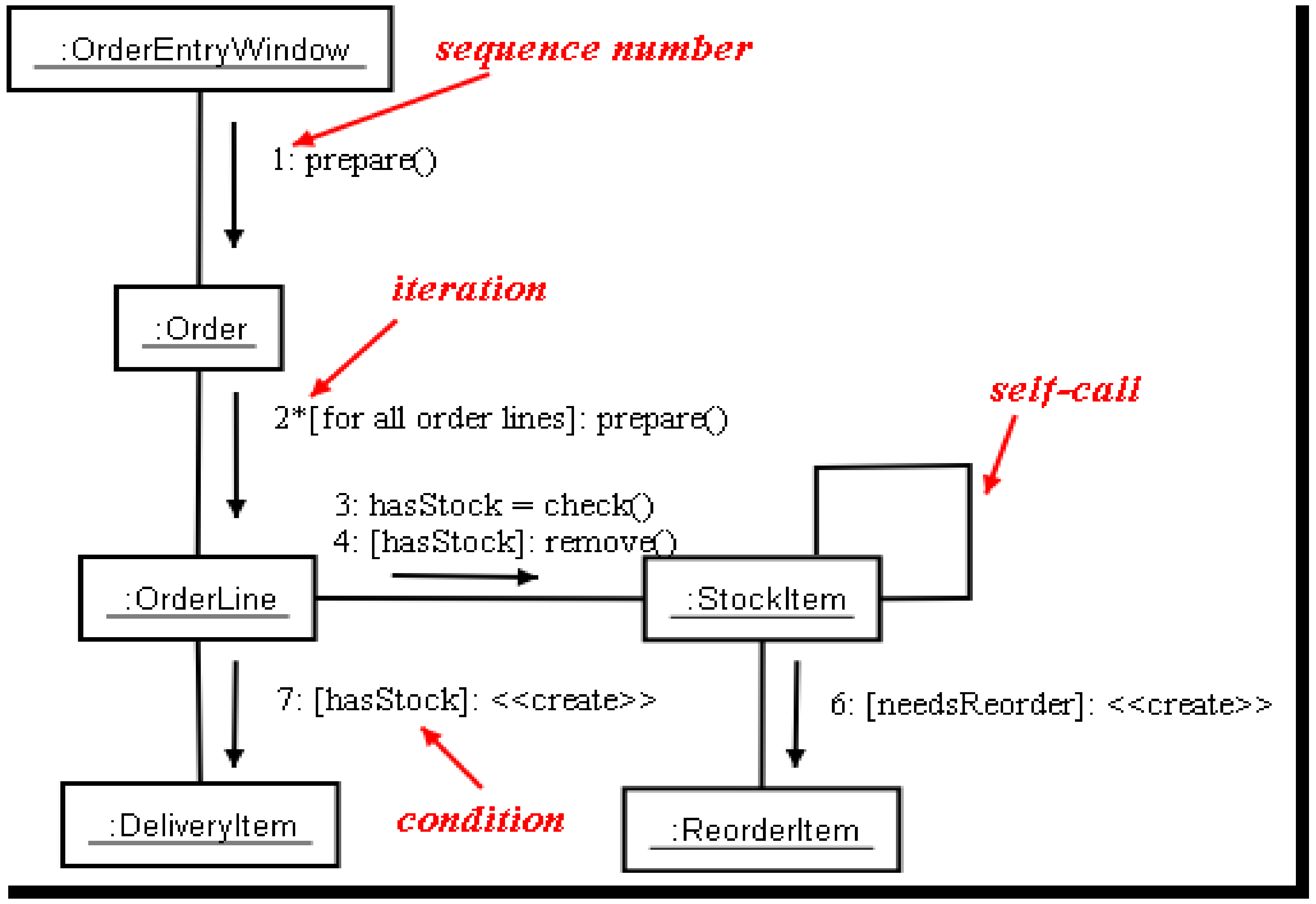
- Use Case diagram（用例图）
- Class diagram（类图）
- Object diagram（对象图）
- Sequence diagram（序列图）
- Collaboration diagram（协作图）
- State diagram（状态图）
- Activity diagram（活动图）
- Component diagram（组件图）
- Deployment diagram（部署图）

Collaboration Diagram

- Collaboration Diagrams show us how objects collaboration in message sequence to satisfy the functionality of a use case.



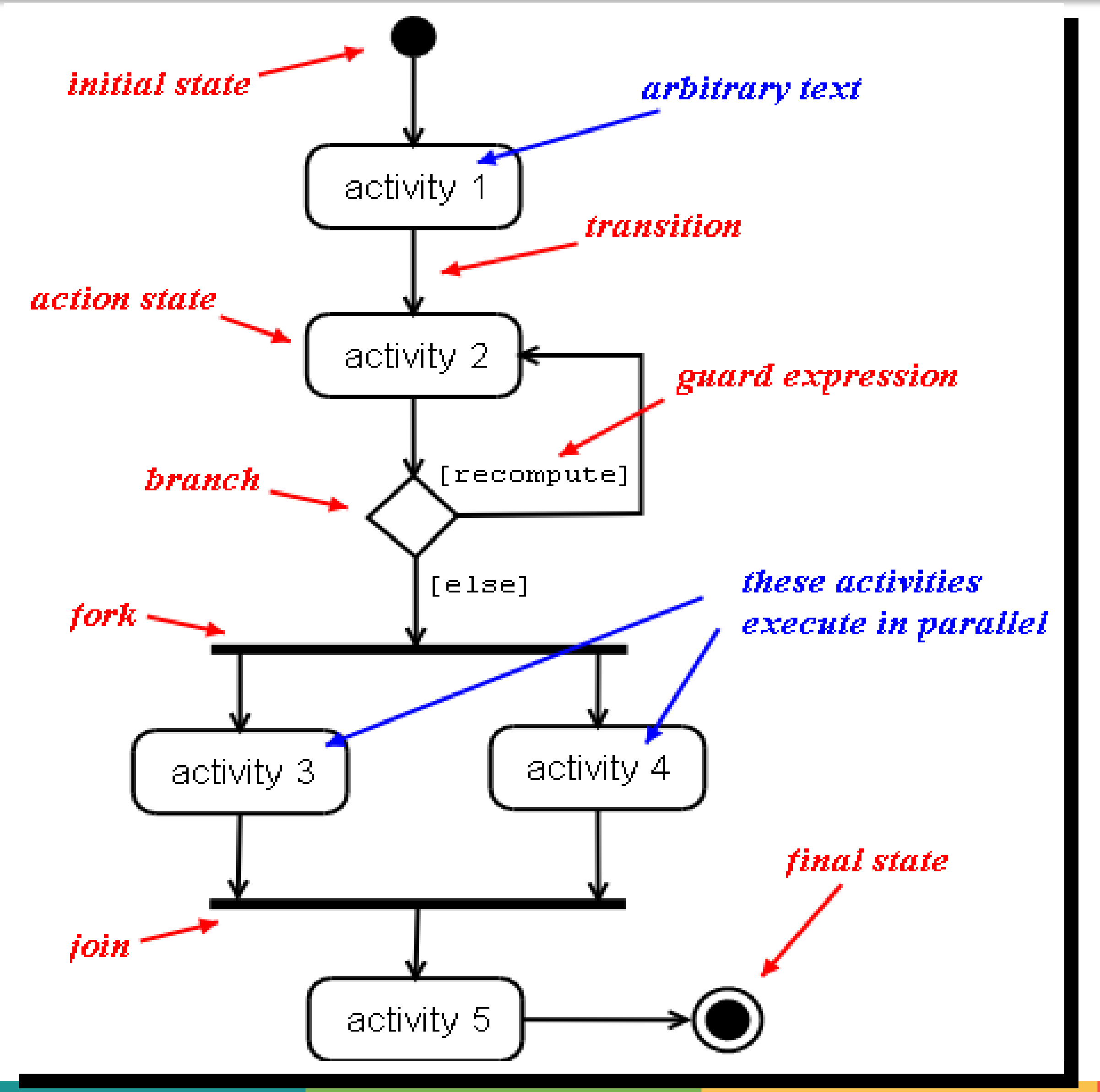
Sample Collaboration Diagram



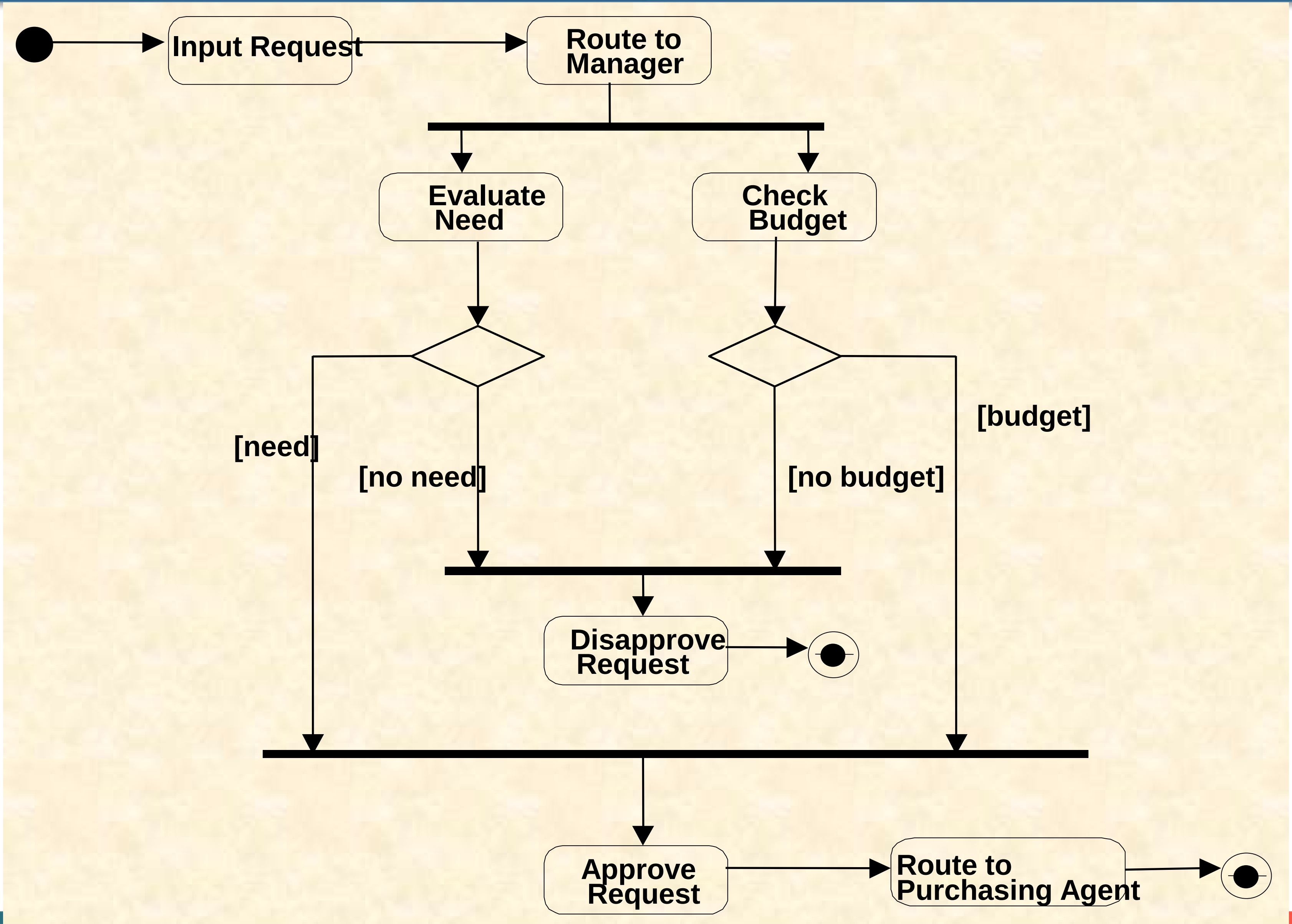
Activity Diagram

- Activity Diagrams are similar to flowcharts in that they graphically depict the sequential flow of activities of either a business process or a use case.
- They are different from flowcharts in that they provide a mechanism to depict activities that occur in parallel.

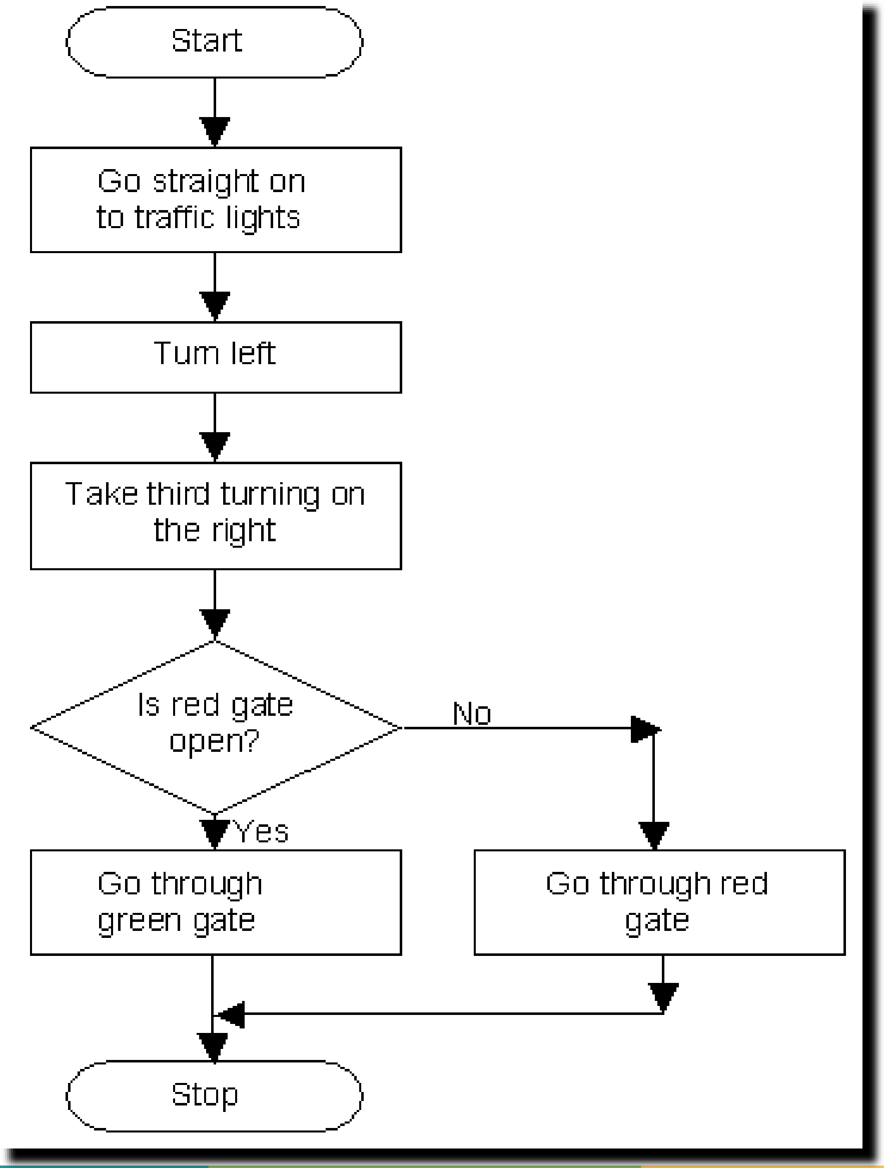
Activity Diagram



Sample Activity Diagram

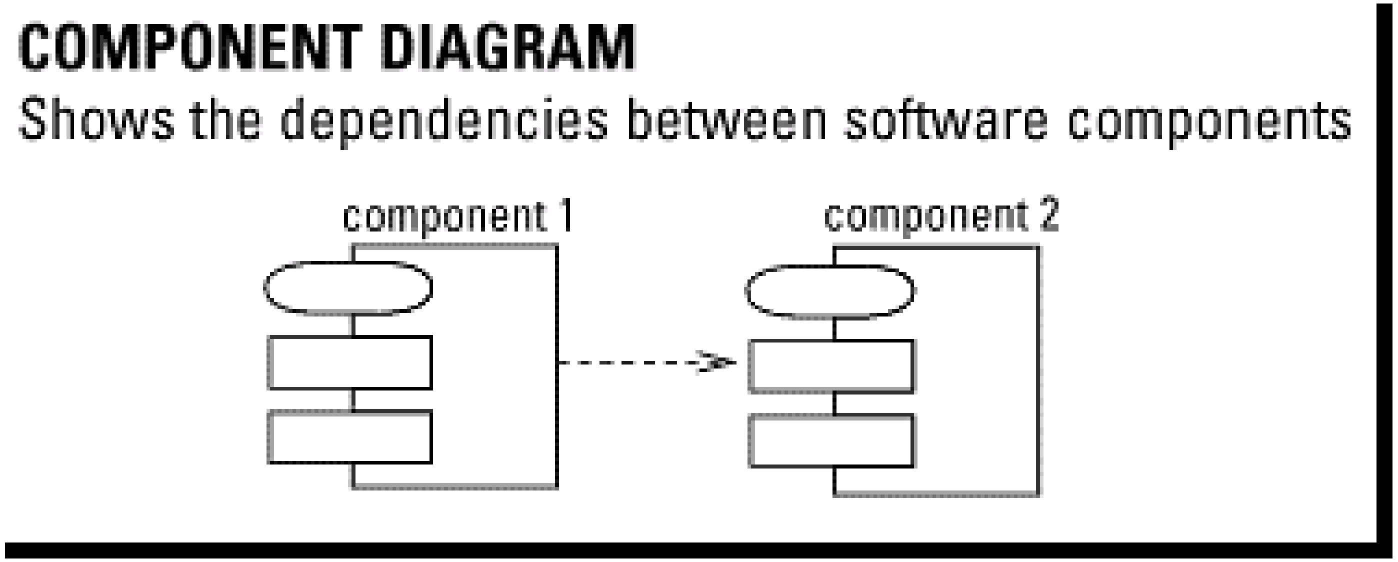


Sample Flowcharts

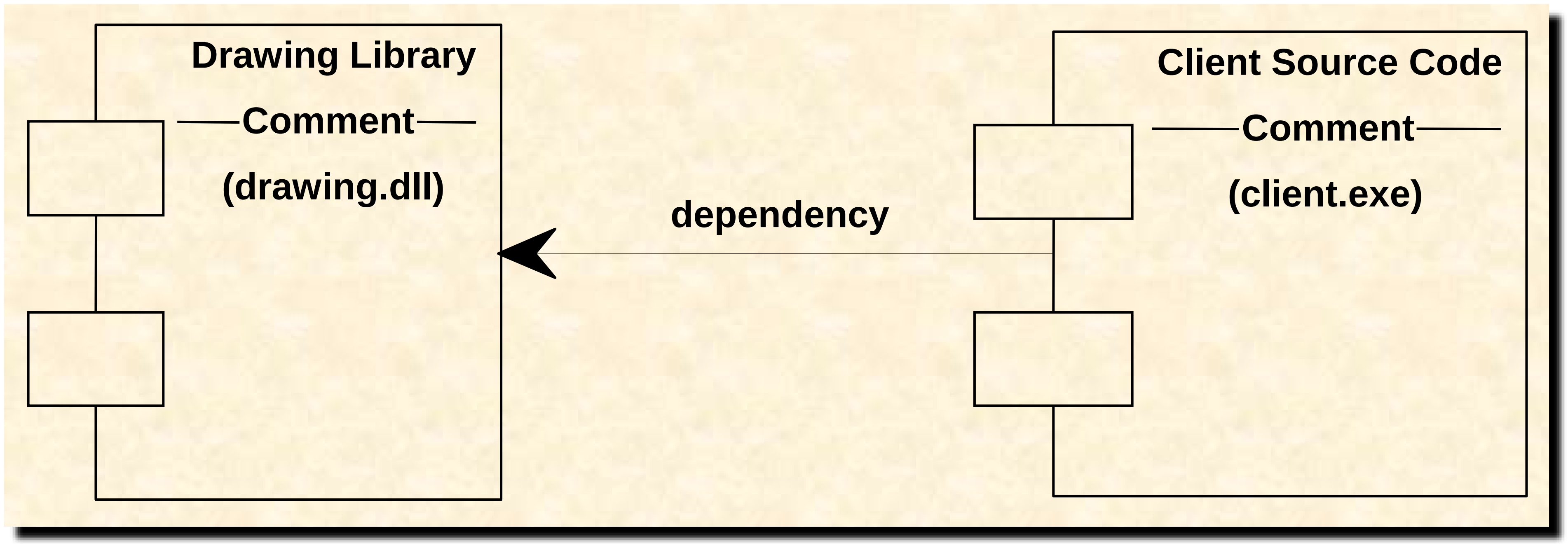


Component Diagram

- Component Diagrams are implementation type diagrams and are used to graphically depict the physical architecture of the software of the system.

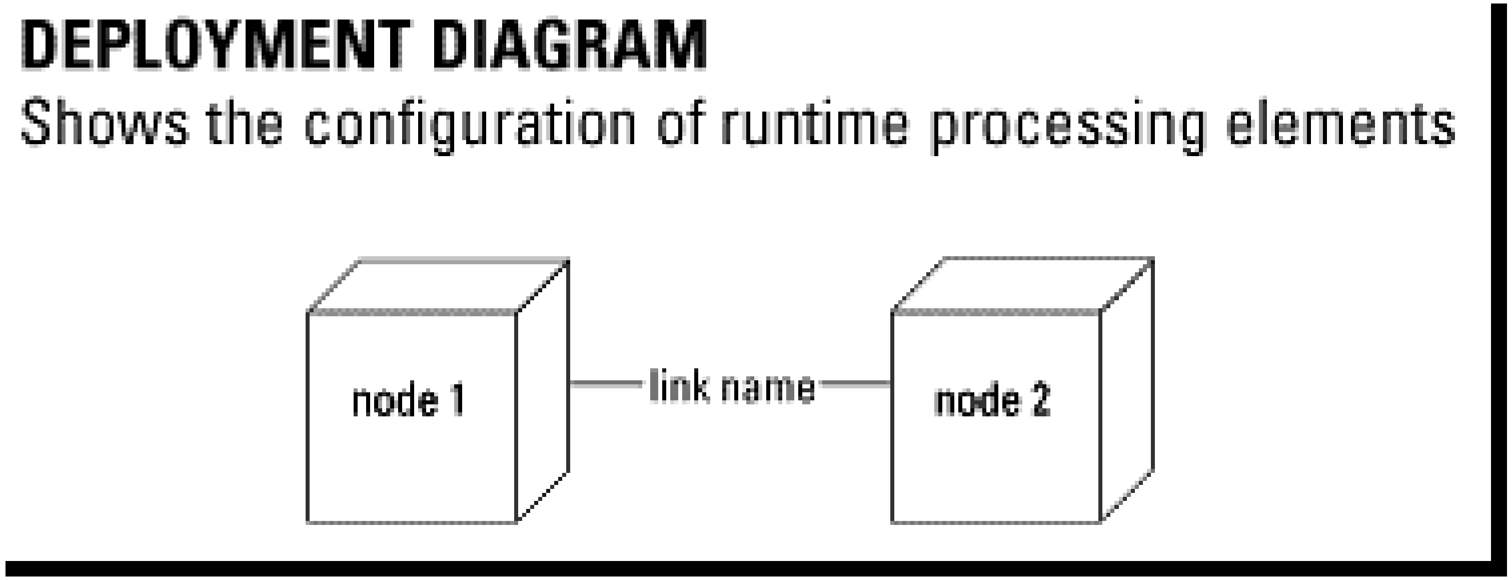


Sample Component Diagram

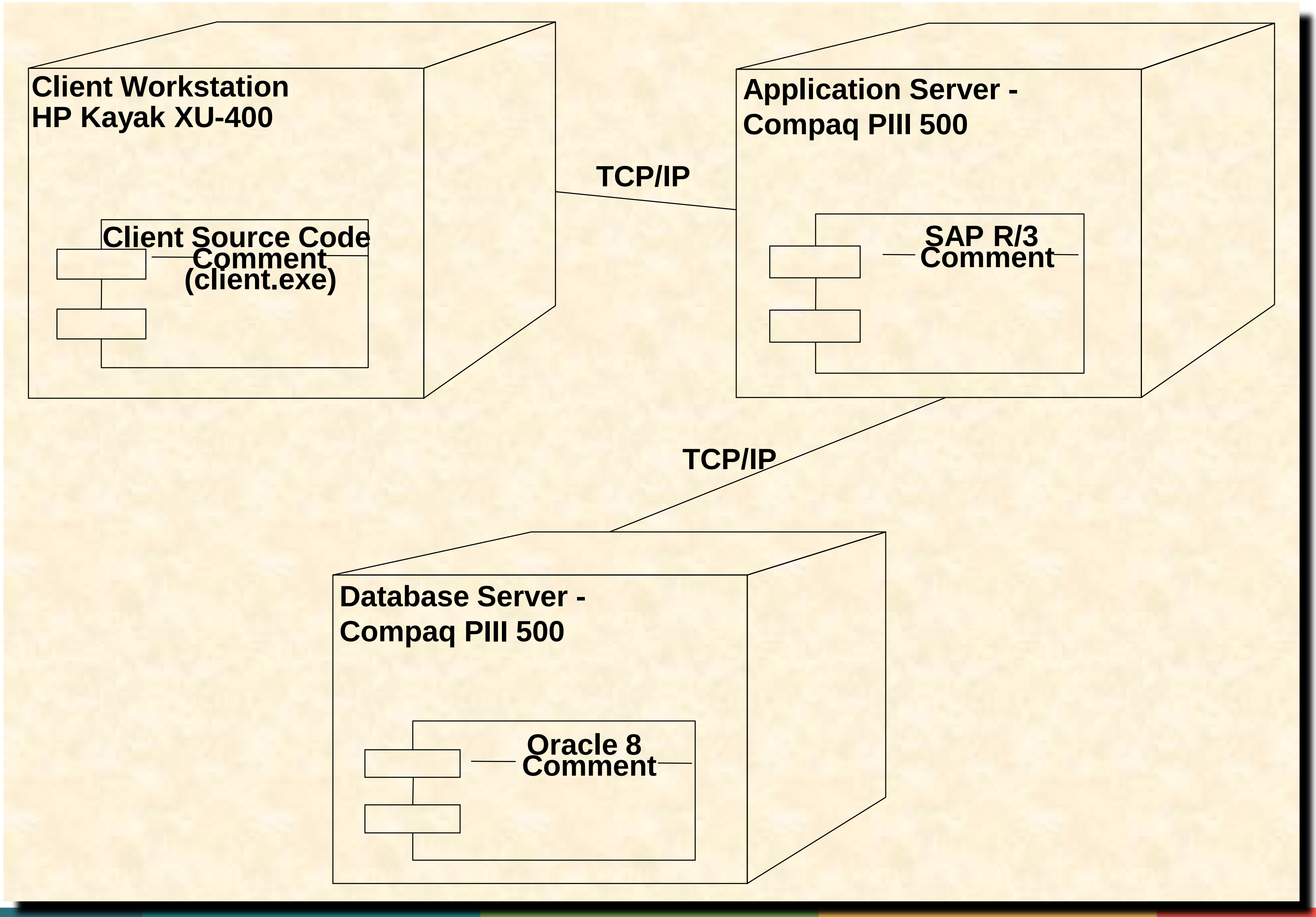


Deployment Diagram

- Deployment Diagrams are also implementation type diagrams that describe the physical architecture of the hardware and software in the system.



Sample Deployment Diagram



Object-Oriented Analysis and Design

- Requirement Analysis
- Use Case Modeling
- Static Modeling
- System Design
- Object Design
- Dynamic Modeling
- Physical Design

Books Management System

Requirement Analysis

在图书管理系统中，管理员要为每个读者建立借阅账户，并給读者发放不同类别的借阅卡（借阅卡可提供卡号、读者姓名），账户内存储读者的个人信息和借阅记录信息。持有借阅卡的读者可以通过管理员（作为读者的代理人与系统交互）借阅、归还图书，不同类别的读者可借阅图书的范围、数量和期限不同，可通过互联网或图书馆内查询终端查询图书信息和个人借阅情况，以及续借图书（系统审核符合续借条件）。

借阅图书时，先输入读者的借阅卡号，系统验证借阅卡的有效性和读者是否可继续借阅图书，无效则提示其原因，有效则显示读者的基本信息（包括照片），供管理员人工核对。然后输入要借阅的书号，系统查阅图书信息数据库，显示图书的基本信息，供管理员人工核对。最后提交借阅请求，若被系统接受则存储借阅纪录，并修改可借阅图书的数量。归还图书时，输入读者借阅卡号和图书号（或丢失标记号），系统验证是否有此借阅纪录以及是否超期借阅，无则提示，有则显示读者和图书的基本信息供管理员人工审核。如果有超期借阅或丢失情况，先转入过期罚款或图书丢失处理。然后提交还书请求，系统接受后删除借阅纪录，并登记并修改可借阅图书的数量。

图书管理员定期或不定期对图书信息进行入库、修改、删除等图书信息管理以及注销(不外借)，包括图书类别和出版社管理。

Books Management System

Use Case Modeling

用例名称：借书

参与的执行者：管理员

前置条件：一个合法的管理员已经登录到这个系统

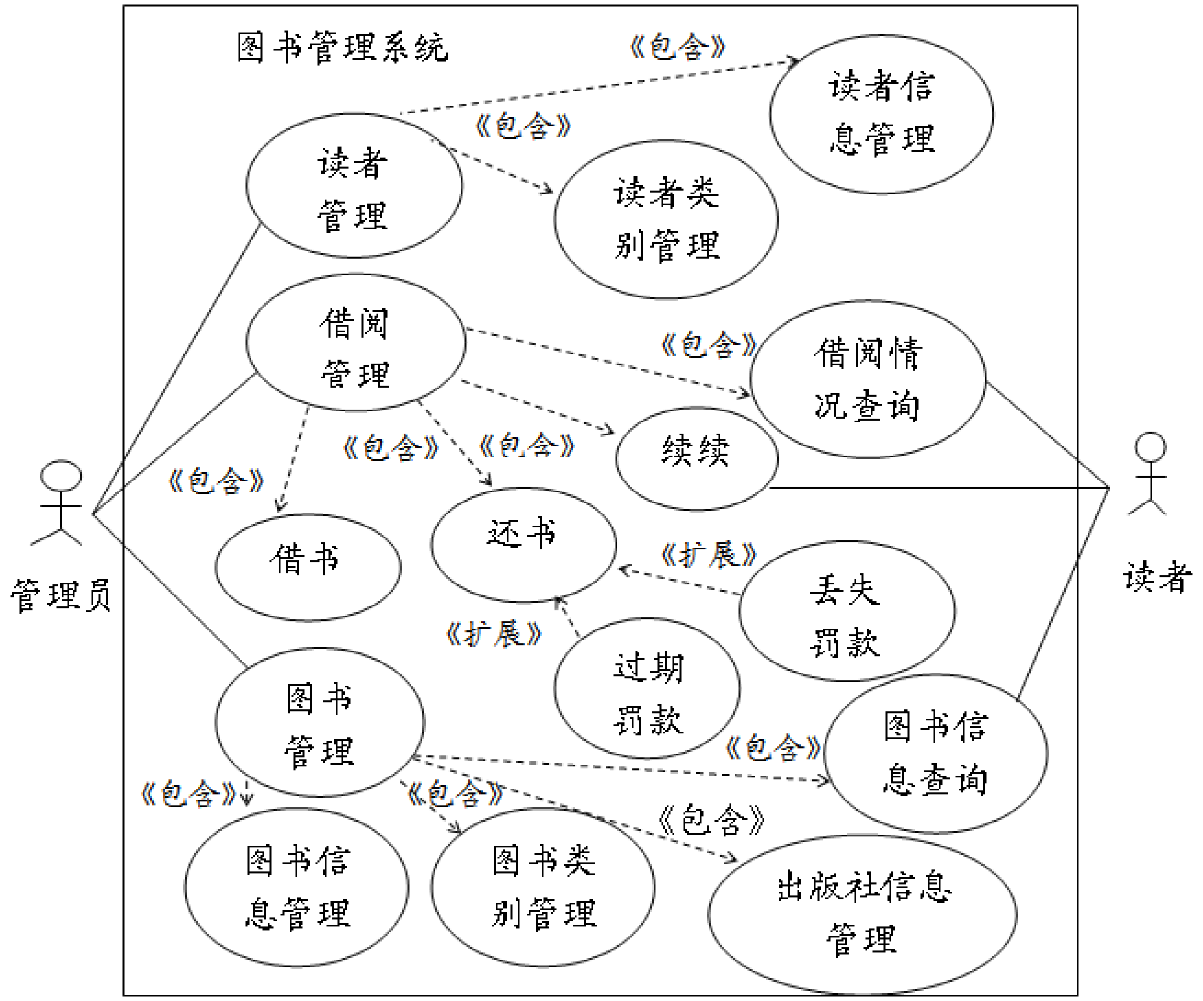
事件流:

```
A.输入读者编号；提示超期未还的借阅记录;
B.输入图书编号;
  If 选择“确定” then
     If  读者状态无效 或 改书“已”注销 或 已借书数>=可借书数 Then  给出相应提示;
    Else
      添加一条借书记录；“图书信息表”中“现有库存量”-1; “读者信息表”中“已借书数量”＋1;
      提示执行情况;
    Endif
    清空读者、图书编号等输入数据;
Endif
If 选择“重新输入”then 清空读者、图书编号等输入数据;
Endif
If 选择“退出”then 返回上一级界面;
Endif 返回 A.等待输入下一条;
```

后置条件：如果是有效借书，在系统中保存借阅纪录，并修改图书库存量和读者借书数量。

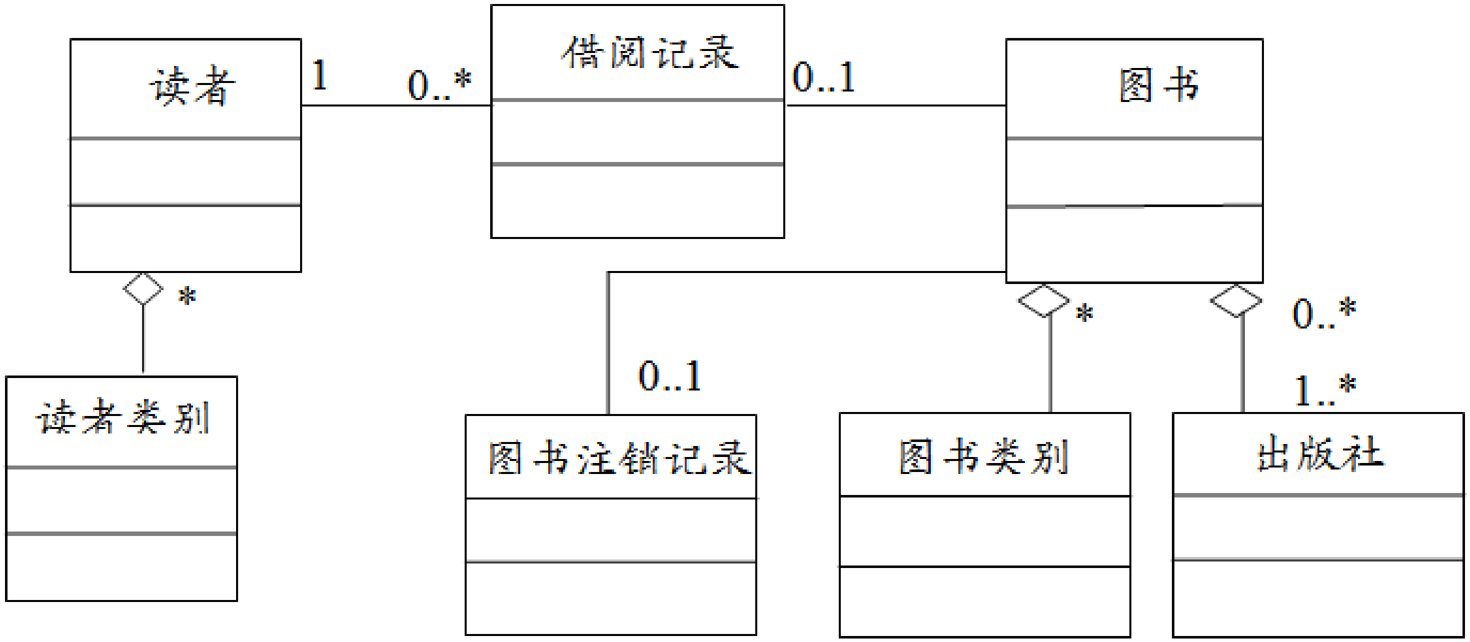
Books Management System

Use Case Modeling



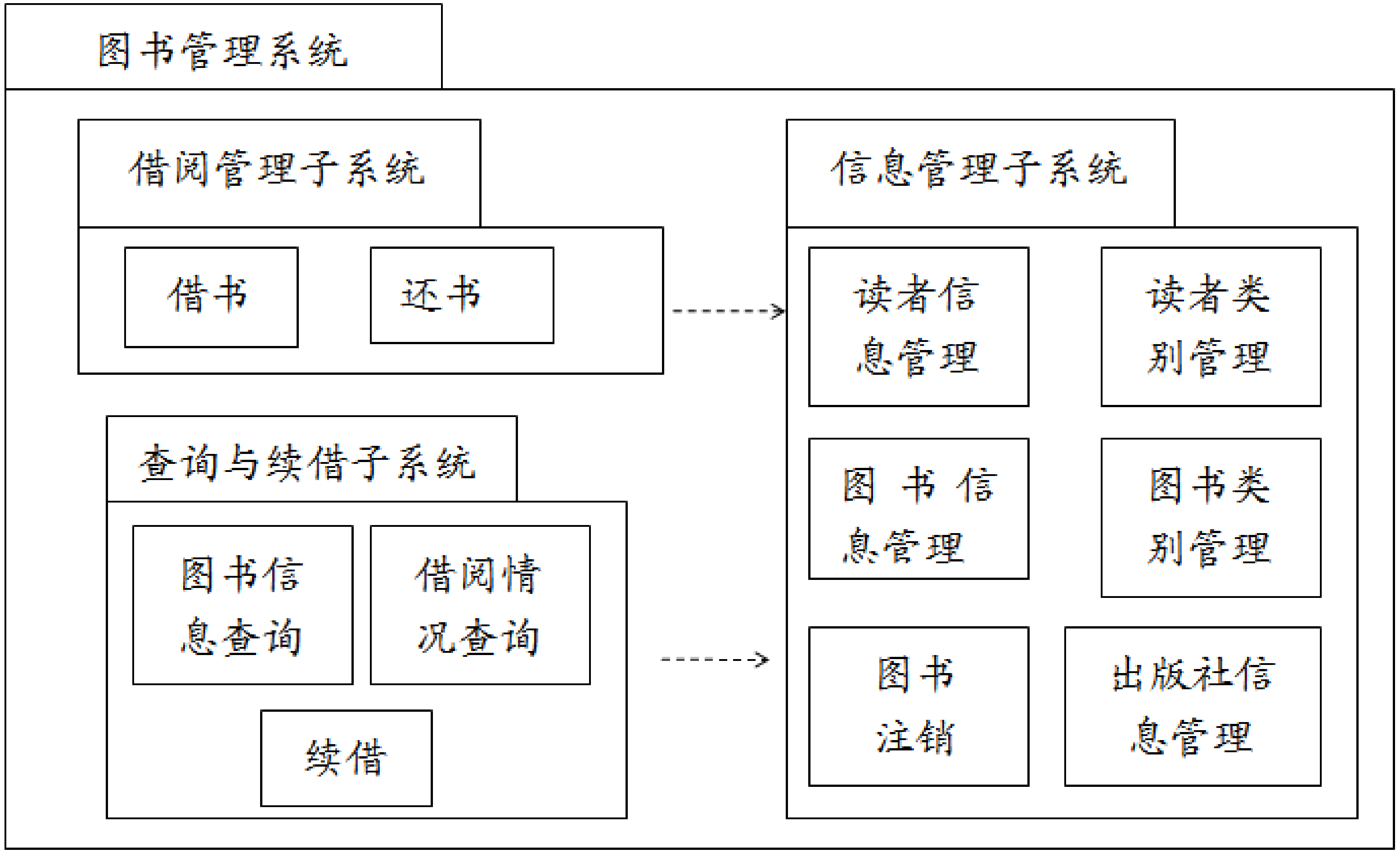
Books Management System

- Static Modeling



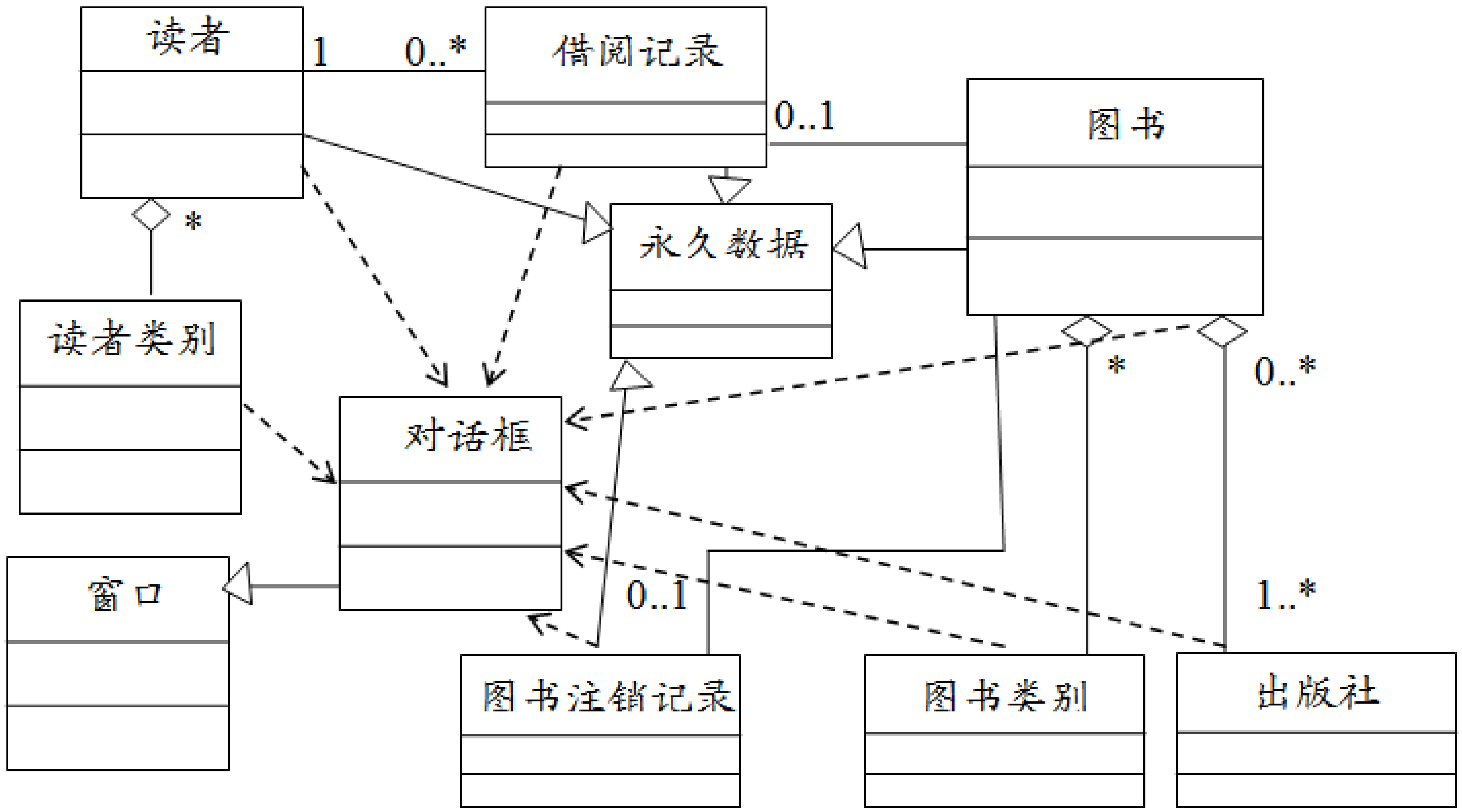
Books Management System

System Design



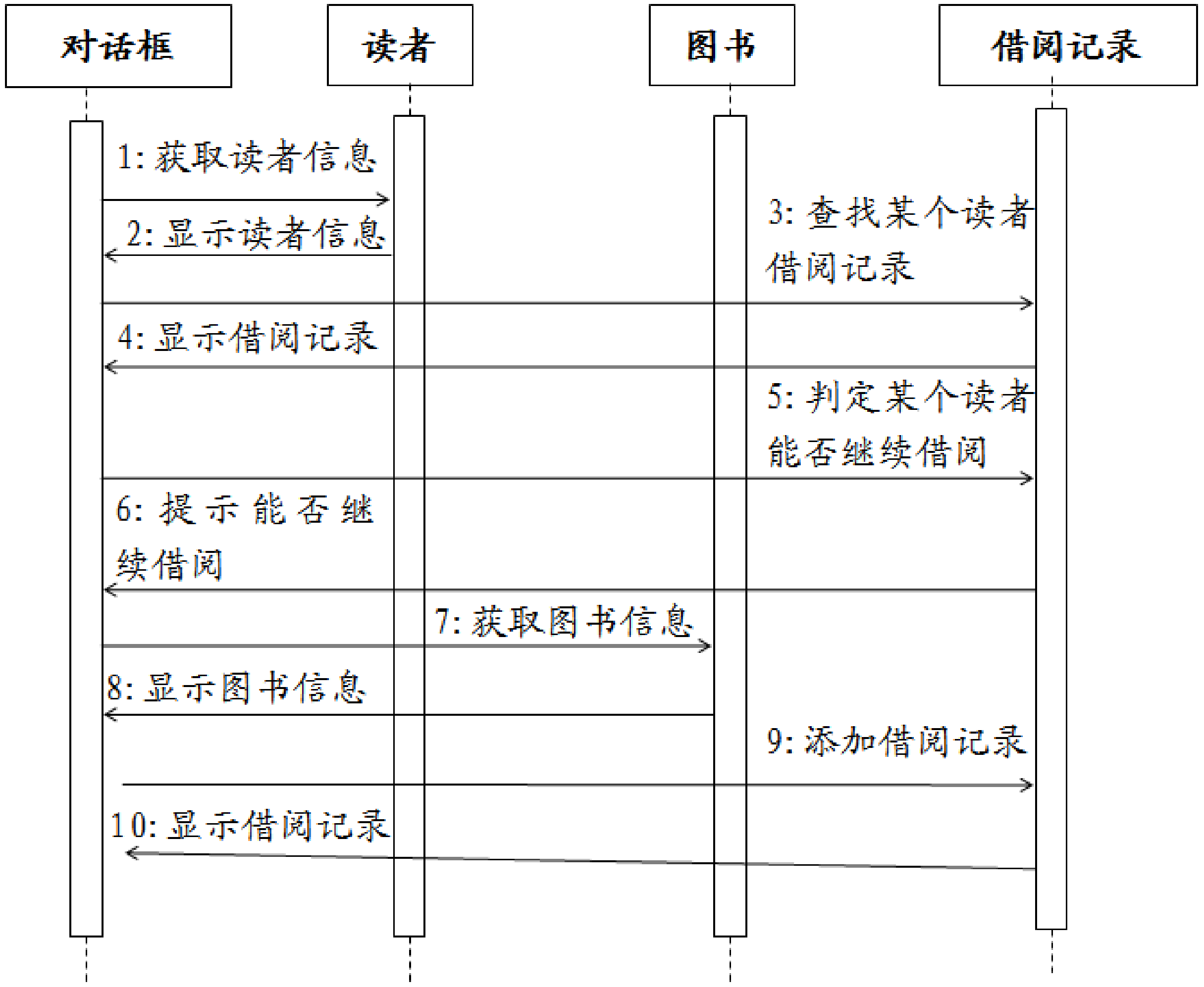
Books Management System

System Design



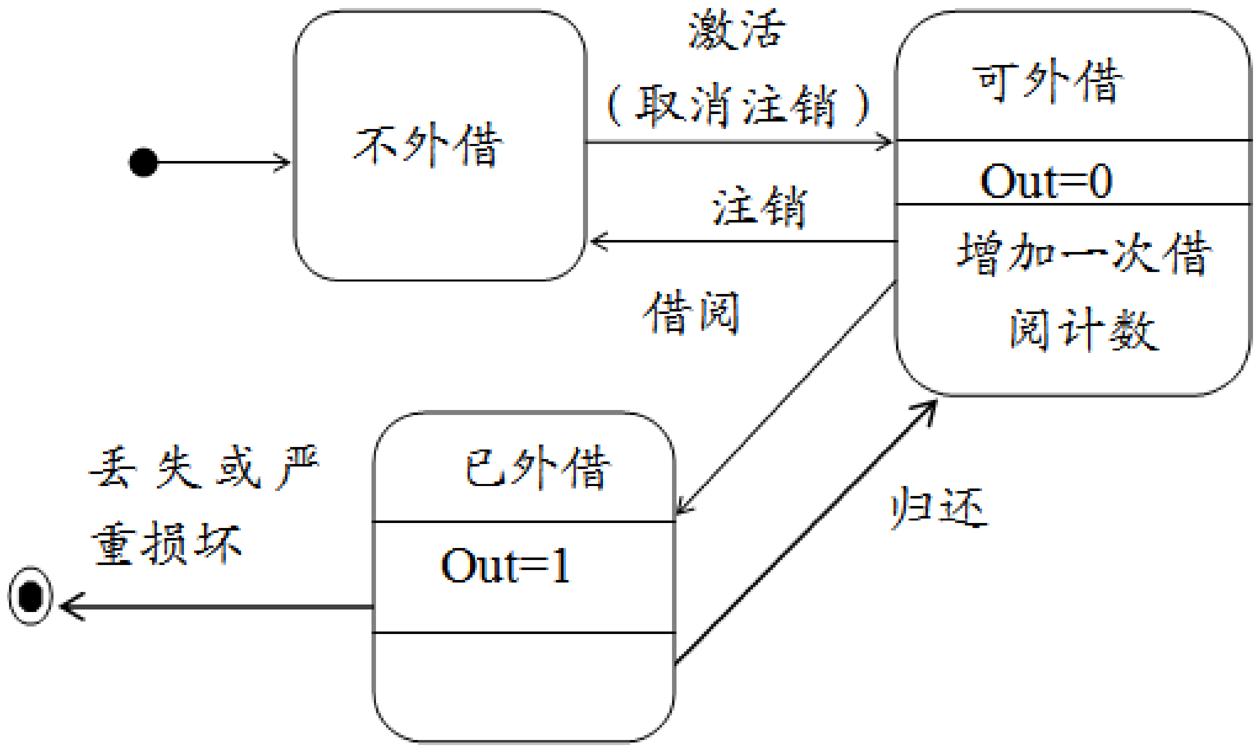
Books Management System

- Dynamic Modeling: Sequence Diagram



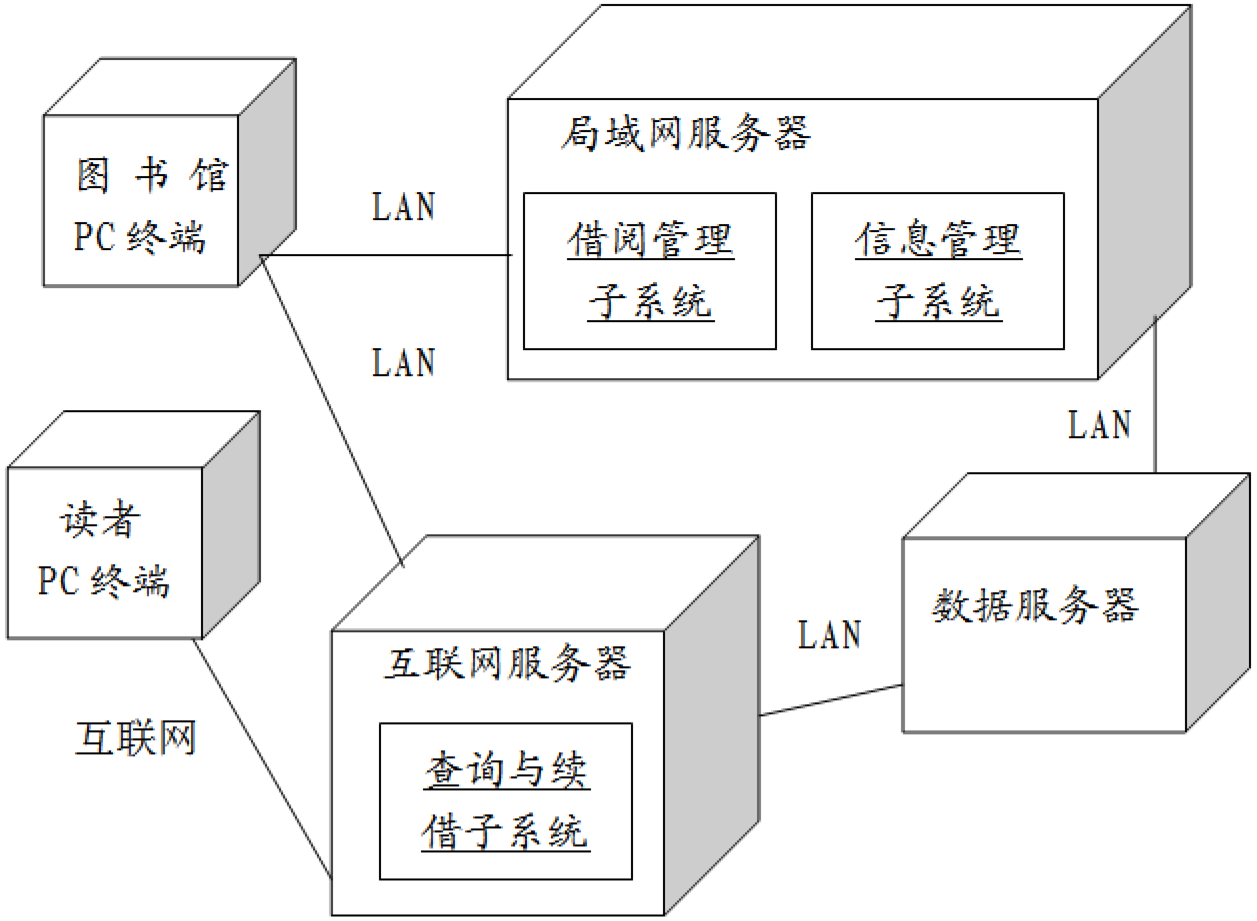
Books Management System

Dynamic Modeling: State Diagram



Books Management System

Physical Design



要点与引申

- 面向对象设计是对面向对象分析的结果进行加细与求精的过程，而用例则是贯穿其中的。
- 分清三类对象的责任。责任涉及：
  - What “I” know? - 每一个对象知道关于它自己的事情；
  - Who “I” know? - 每一个对象知道有关的其他对象；
  - What “I” do? - 每一个对象知道它必须履行的功能。
- 从设计结果可以看出，这里的“对象”在经历了选择和评估之后，被映射成了程序设计时的类。