Object Oriented Design and Modeling through UML

Chapter-2

Object Oriented Analysis

Facilitator: Er. Rudra Nepal

Object Oriented Analysis

- Here, we deal with the investigation of the problem that is associated with the system i.e. we deal with ‘what’ type of questions **what the problem is about** and **what the system must do.**

Building Domain Model/Conceptual Model

- Domain models are the static structure diagrams in which no operations are defined .
- It is the representation of the real-world conceptual classes, not of the software components .
- It may show:
  - **domain objects or conceptual classes**
  - **associations between the conceptual classes**
  - **attributes of those conceptual classes**

Building Domain Model/Conceptual Model

- It must not show:
  - **software artifacts such as database**
  - **responsibilities or methods**
- Domain models are created to get a better understanding of the key concepts of the system. Also, it helps in reducing the gap between software representation and the conceptual model of the particular system.

Building Domain Model/Conceptual Model

- There are three steps of creating a domain model :
  1. **Identifying the domain concepts or conceptual classes**
  2. **Adding associations**
  3. **Adding attributes**

1. Identifying the domain objects or conceptual class

- Conceptual class is an idea or thing that is related to a system .
- Conceptual class can be considered in terms of symbol, intension and extension.
  - **Symbol** : word or image representing a class
  - **Intension**: definition of a particular class
  - **Extension** : the set of examples that applies to a conceptual class or objects

1. Identifying the domain objects or conceptual class

- Strategy to identify the conceptual class :
  a. **Use the conceptual class category list**
  b. **Identify the noun or noun phrase**

a. Use the conceptual class category list

- Candidate conceptual class can be identified as :

Conceptual class category	Example
-Business transactions (involving money)	-Sale, Payment
-Transactions line items	-SalesLineItem
-Service related to a transaction	-Item
-Where is the transaction recorded?	-Register
-People who have roles in transactions or actors in use cases	-Customer, Cashier

a. Use the conceptual class category list

Conceptual class category	Example
-Place of transaction or place of service	-Store
-Physical or tangible objects	-Item
-Specification or description of things	-ProductSpecification
-Catalog	-ProductCatalog
And so on.	

b. Identify noun or noun phrases

- Here, we identify noun or noun phrases in the textual description of the system and consider them as a conceptual class.
- For example,
  - **Main success scenario of the POS system:**
    1. **Customer** arrives at the **POS checkout** with **goods**.
    2. **Cashier** starts a new **sale**.
    3. **Cashier** enters the **item identifier code.**
    4. **System** records **sale line item** and presents the **item description**, **price** and running **total**. **Cashier** repeats the step 2 and 3 until all the **goods** are entered .

b. Identify noun or noun phrases

5. System presents the **total** with **taxes** calculated.
6. **Cashier** tells **customer** the **total** and asks for **payment.**
7. **Customer** pay and system handles the **payment.**
8. System logs the completed **sale** and sends the sale and **payment** info to the external **accounting** and the **inventory** systems to update the **inventory.**
9. System presents **receipts.**
10. **Customer** leaves with **receipt** and **goods.**

b. Identify noun or noun phrases

➢ **Alternate scenario flow for POS system :**

- If the **customer** pays by **cash**, then the **customer** pays the **cash amount** to the **cashier** .
- And so on.

b. Identify noun or noun phrases

- After filtering the classes obtained from two different strategies, we now obtain the following conceptual classes:

Sale | **Customer** | **Cashier** | **Payment** | **Item**

Register | **SaleLineItem** | **ProductSpecification** | **ProductCatalog**

Store

Common mistakes that are made while identifying the conceptual classes

- To represent an idea as an attribute which has to be represented as a conceptual class.
- The common rule :
  - **If we do not think of some conceptual class X as a number or text in a real world, then X is probably a conceptual class, not an attribute.**
  - **For example,**
    - **Should the store be an attribute of sale or a different conceptual class?**
      - **Here, the store cannot be thought of as an integer number or text in the real world and it occupies some space. Therefore, it should be a different conceptual class.**

2. Adding associations between the conceptual classes

- Types of associations
- Multiplicity

3. Adding associations between the conceptual classes

- Associations can be identified as:

Category	Example
A is a physical part of B	Register –CashDrawer
A is a logical part of B	SaleLineItem –Sale
A is physically contained in B	Register –Store, Item –Store
A is logically contained in B	ProductSpecification –ProductCatalog, ProductCatalog –Store
A is a description of B	ProductSpecification-Item
A is a line item of a transaction or A is logged in B	Sale –Store (when the transaction is complete), Sale –Register (when the transaction is incomplete)

2. Adding associations between the conceptual classes

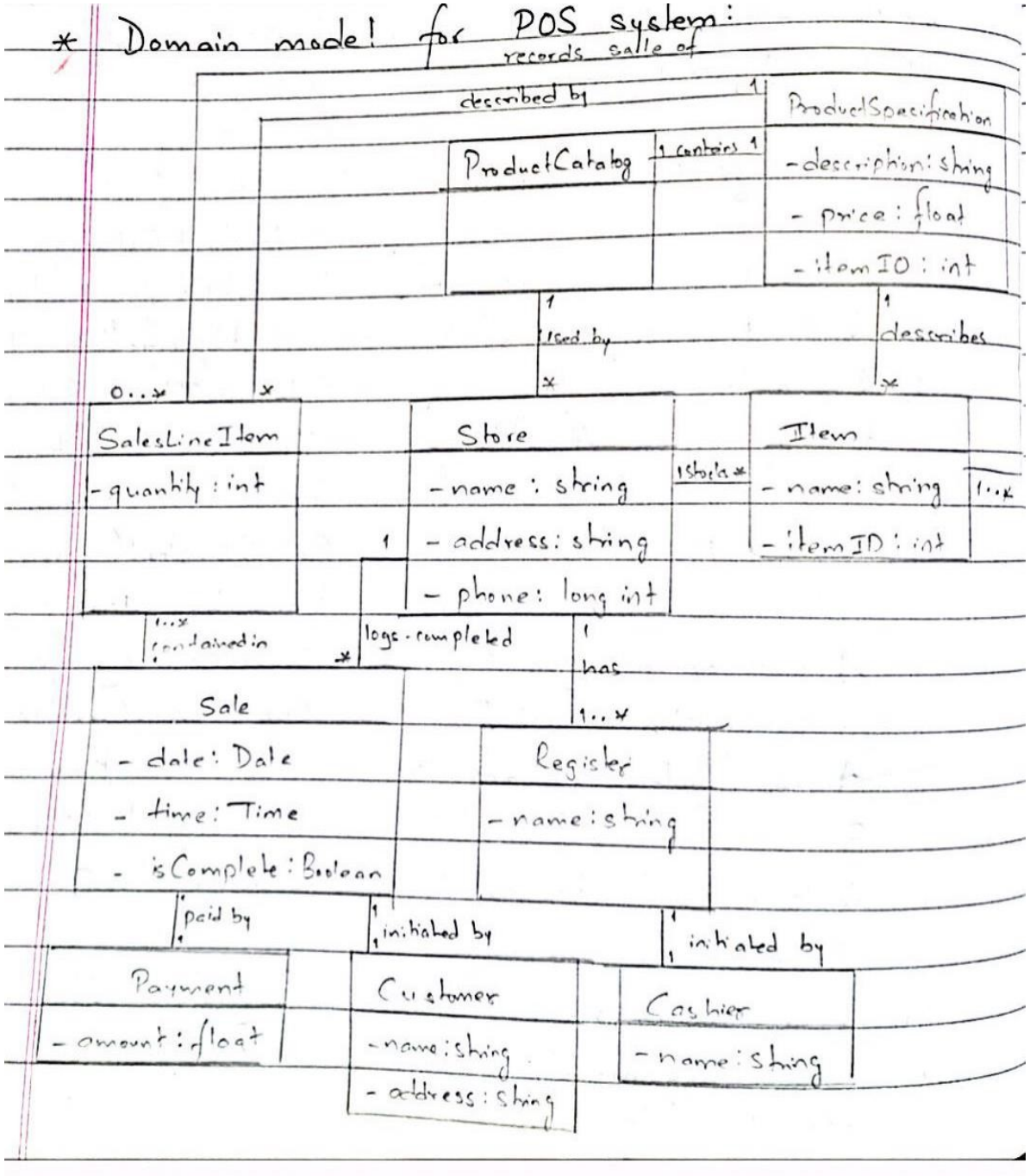
- Associations can be identified as:

Category	Example
A uses or manages B	Cashier-Register
A communicates with B	Customer –Cashier
A is related to transaction B	Customer –Payment, Cashier –Payment
A is a transaction related to another transaction B	Payment –Sale
A is owned by B	Register –Store
A is an organizational subunit of B	not applicable

3. Adding Attributes

- Attributes are any physical entities that are related with the conceptual class. They are shown inside the second compartment of the class.
- **For example, for the class sale, the attributes can be date, time, quantity and so on.**

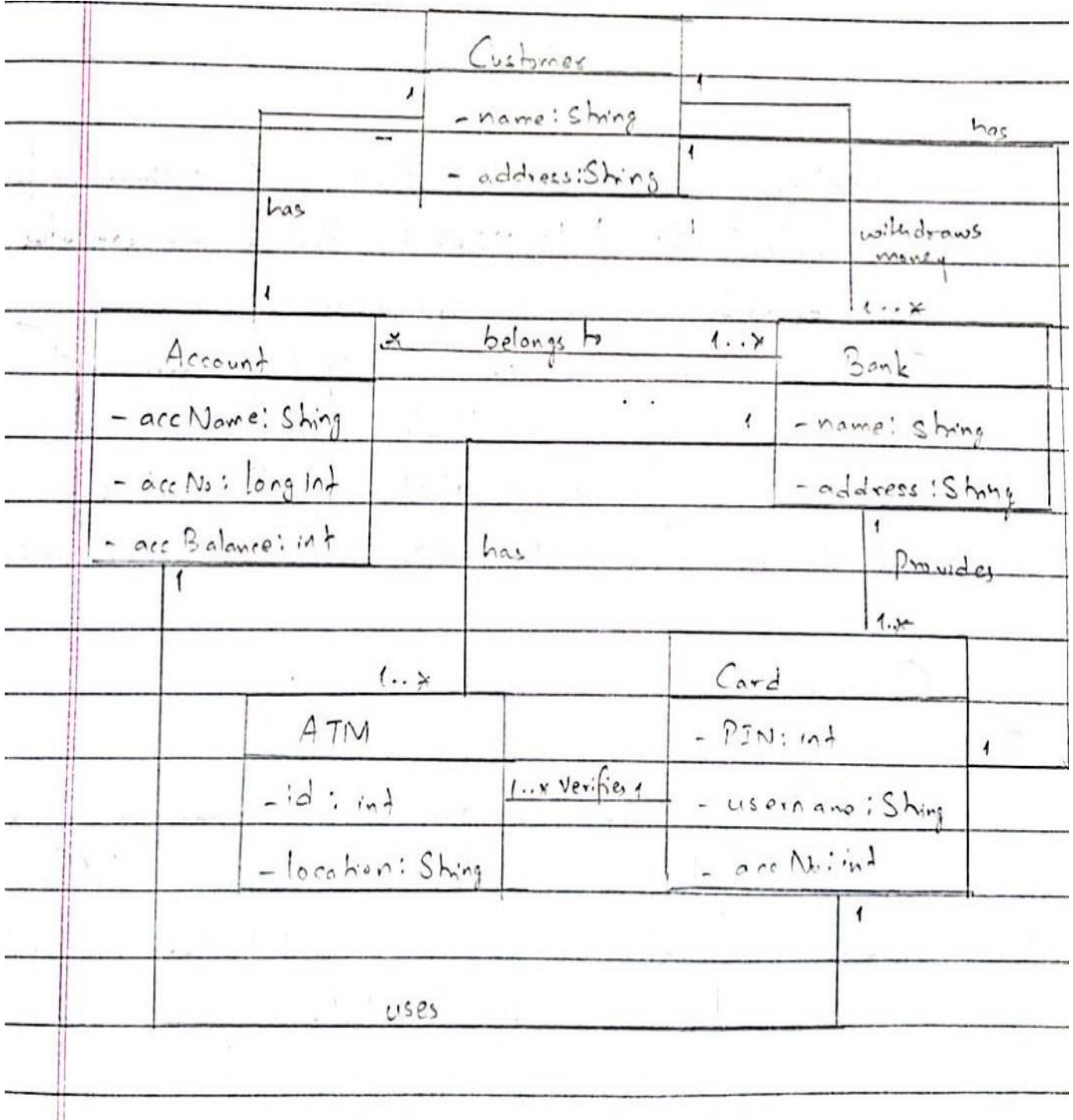
Domain model for POS system



Case study

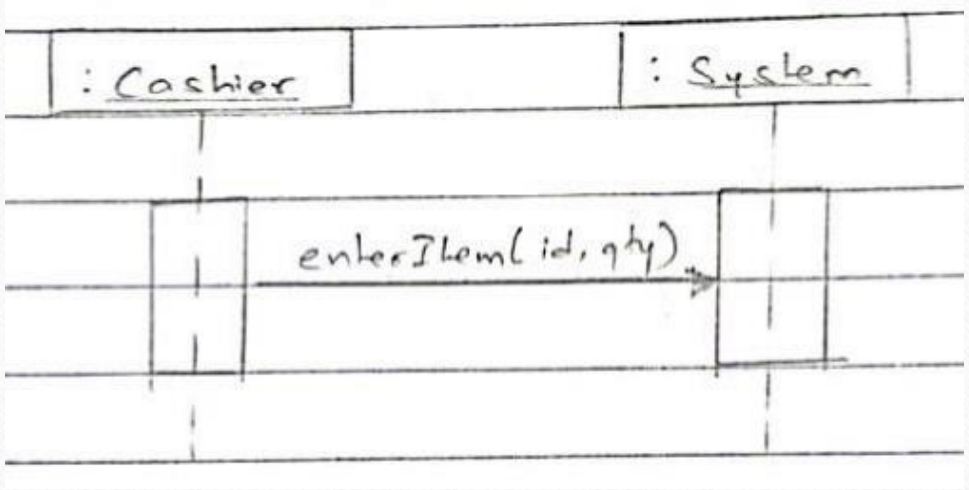
- A customer wants to draw money from the bank account. He enters his card into an ATM. The ATM machine prompts ‘Enter pin’. The ATM internally retrieves the bank account number from the card. The ATM encrypts the pin and the account number and sends it to the bank. The bank verifies the encrypted account and pin number. If the pin number is correct, the ATM displays ‘Enter amount’ , draws money from the bank account and pays out the amount. Draw the domain model for the above case study.

Domain model for ATM Machine



System event and System operation

- **System event** is an external input generated by a user to a system.
- Certain operation is performed by the system in response to the system event. Such kind of operation is known as **System operation**.



System event and System operation

- **In the above example, when a cashier performs the ‘Enter item’ system event, the system executes the ‘Enter item’ system operation .**
- Therefore , the names of the system event and system operation can be identical.
- The only difference is that system event is a stimuli and a system operation is the response.

Contract

- Use cases are the primary mechanism to describe the system behavior and are usually sufficient. However, sometimes a more detailed description of system behavior is needed.
- Contracts describe the detail system behavior in terms of the state changes of the objects in domain model after the system operation has been executed.
- It is document that describes what an operation commits to achieve.

Contract

- It is generally declarative in style emphasizing what will happen rather than how it will be achieved.
- Contracts are expressed in terms of pre-conditions and post-conditions.
- A contract can be written for an individual method of a software class or for an individual system operation.

Format of a Contract

- **Operation:**
  - ➢ It is the name of the operation for which contract is being built.
- **Cross Reference :.**
  - ➢ It is the name of the use case in which the particular system operation occurs.
- **Pre-condition :**
  - ➢ It is the condition that must be true prior to the execution of system operation.
- **Post-condition :**
  - ➢ It is also known as **success guarantee**. Each indicates the state changes after the successful completion of a system operation.

Pre-condition

- It defines the assumptions about the state of the system at the beginning of the operation. Pre-condition is built on the basis of:
  - **things that are important to test in software at some point during the execution of the operation,**
  - **things that will not be tested but upon which the success of an operation depends.**

Post-condition

- Post-condition simply represents the state changes and can be divided into three categories:

1. **Instance creation or deletion:**

   ➢ It deals with what new object of a class would have been created after the successful completion of operation.

   ➢ **For example,**

   For POS application, after the Enter item operation has been performed, a SaleLineItem instance should have been created.

Post-condition

2. **Attribute modification :**

   - It deals with what attributes of new or existing objects should have been modified after the successful completion of an operation.
   - **For example,**

     For POS application, after the Enter item operation has been performed, the quantity attribute of SaleLineItem should be equal to the quantity entered by the cashier.

Post-condition

3. **Association formed or broken :**

   - it deals with what association between the classes should have been created after the successful completion of an operation. .
   - **For example,**

For POS application, after Enter item operation has been performed, the SaleLineItem should have been associated to the Sale and ProductSpecification.

Guidelines for building a contract

- To make a contract :
  - ➢ Identify the system operations from the system sequence diagram .
  - ➢ For the system operation data complex to understand, construct a contract.
  - ➢ For the post conditions, use the following categories:
    - ❑ **Instance creation or deletion**
    - ❑ **Attribute modification**
    - ❑ **Association formed or broken**

Advice on writing a contract

- State the post conditions in a declarative passive past tense form to emphasize the declaration of state change rather than the design of how it is going to be achieved.
- **For example,**
  - **A SaleLineItem instance was created.**
- Remember to add association between the existing object and the newly created ones.

Examples for contracts of a POS system

- **Contract CO1** : Make a new sale
- **Operation :** makeaNewSale()
- **Cross Reference:** Use case-POS
- **Pre-condition:**
  - Cashier should be verified.
- **Post-conditions:**
  - A Sale instance S was created. (Instance creation)
  - S was associated with Register.(Association formed)
  - S.date and S.time was initialized by cashier. (Attribute modification)

Examples for contracts of a POS system

- **Contract CO2** : Enter Item
- **Operation :** enterItem(id:int, quantity:int)
- **Cross Reference:** Use case-POS
- **Pre-conditions:**
  - Cashier should be verified.
  - There must be an undergoing sale.
- **Post-conditions:**
  - A SaleLineItem instance sli was created. (Instance creation)
  - sli.quantity was initialized to quantity entered by the cashier. (Attribute modification)
  - sli was associated with Sale. (Association formed)
  - sli was associated with ProductSpecification based on the id of an item. (Association formed)

Examples for contracts of a POS system

- **Contract CO3** : End Sale
- **Operation :** endSale()
- **Cross Reference:** Use case-POS
- **Pre-conditions:**
  - Cashier should be verified.
  - All items are entered.
- **Post-conditions:**
  - S.isComplete was initialized to true. (Attribute modification)

Examples for contracts of a POS system

- **Contract CO4** : Make Payment
- **Operation :** makePayment()
- **Cross Reference:** Use case-POS
- **Pre-conditions:**
  - Sale must be ended.
- **Post-conditions:**
  - A payment instance P was created. (Instance creation)
  - P.amount was initialized to the total amount calculated, (Attribute modification)
  - P was associated with sale. (Association formed)
  - Sale was associated with Register and Register was associated with Store. (Logs-completed) (Association formed)

Practice Question : Online course registration system

- At the beginning of each semester, students may request a course catalog containing a list of course offerings for the semester. Information about each course such as professor, department and pre-requisites will be included to help students for making an informed decision.
- The new online registration system will allow students to select four course offerings for the coming semester. In addition, each student will indicate 2 alternative choices in case a course offering becomes filled or cancelled.

Practice Question : Online course registration system

- No course offering will have more than 10 students and fewer than three students. A course offering with fewer than three students will be cancelled . Once the registration process is completed for a student, the registration system sends the information to the billing system so that the student can be billed for the semester.
- Professors must be able to access the online system to indicate which courses they will be teaching. They will also need to see which students signed up for their course offering.

Practice Question : Online course registration system

- For each semester, there is a period of time that students can change their schedules. Students must be able to access the online system during this time to add or drop the courses . The billing system will credit all the students for courses dropped during this period of time.
- Prepare a contract for the system operation to register a particular course by the student.

Few more practice questions

- Draw domain model for ATM system and write down a contract for the system operation to withdraw money from ATM.
- Draw domain model for Library Management System and write down a contract for the system operation to return a book.
- Draw domain model for online shopping system and write down a contract for the system operation to make payment after buying an item.
- Draw domain model for hospital management system and write down a contract for the system operation to take appointment.
- Draw domain model for online hotel reservation system and write down a contract for the system operation to book a room.