EC3290 – Software Requirements Engineering

Topic 3 – Requirements Analysis, Modeling, and Specification

Learning Objectives

- Understand the difference between **description and specification**.

- Learn various **modeling techniques and diagrammatic notations**.

- Explore **Use Case Modeling and Scenario Descriptions**.

- Gain insights into **domain analysis and requirements understanding**.

- Introduce **formal specification techniques**.

Introduction to Requirements Analysis

- Process of determining user expectations for a new or modified system.

- Prevents requirement misunderstandings.

- Ensures better system design.

- Reduces project failure risk.

Description vs. Specification

- **Description:**
  - Natural language explanation of system behavior.
  - Often informal and ambiguous.
- **Specification:**
  - Precise and structured representation of system behavior.
  - Uses models, diagrams, and formal notations.
- **Example:**
  - Description: "The system should allow users to login."
  - Specification: "The system shall authenticate users using a valid username and password stored in the database."

Requirements Modelling Techniques

- **Purpose of Modelling:**
  - Visual representation of complex system requirements.
  - Enhances understanding and communication.
- **Key Techniques:**
  - **Use Case Modelling**
  - **Data Flow Diagrams (DFD)**
  - **Entity-Relationship Diagrams (ERD)**
  - **Class Diagrams (UML)**

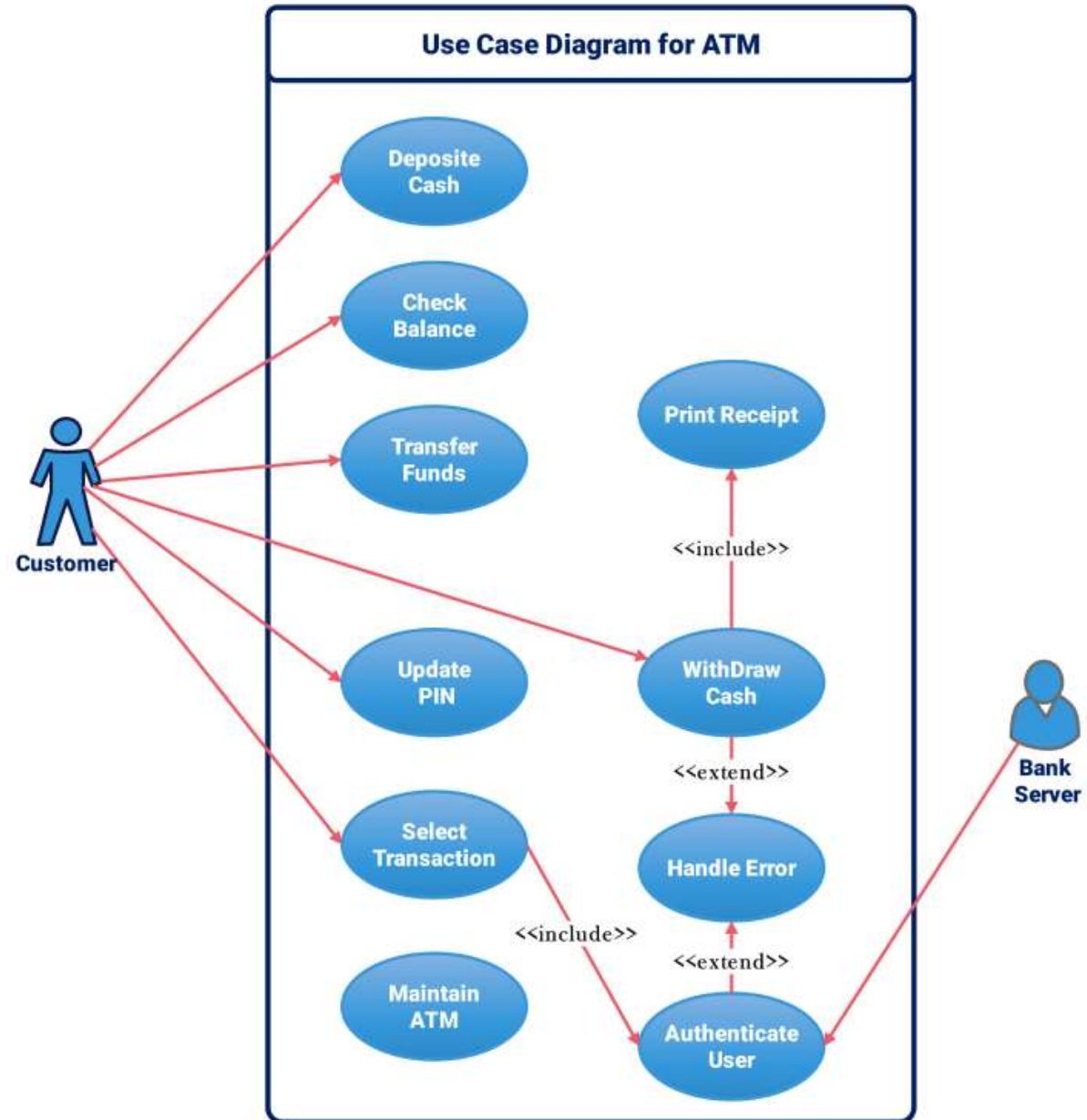# Use Case Modelling and Scenario Descriptions

- **Use Case Diagram:**
  - Captures interactions between users and the system.
  - Identifies system functionality.

- **Scenario Descriptions:**
  - Details different paths in system operation (Normal, Alternative, Exception flows).
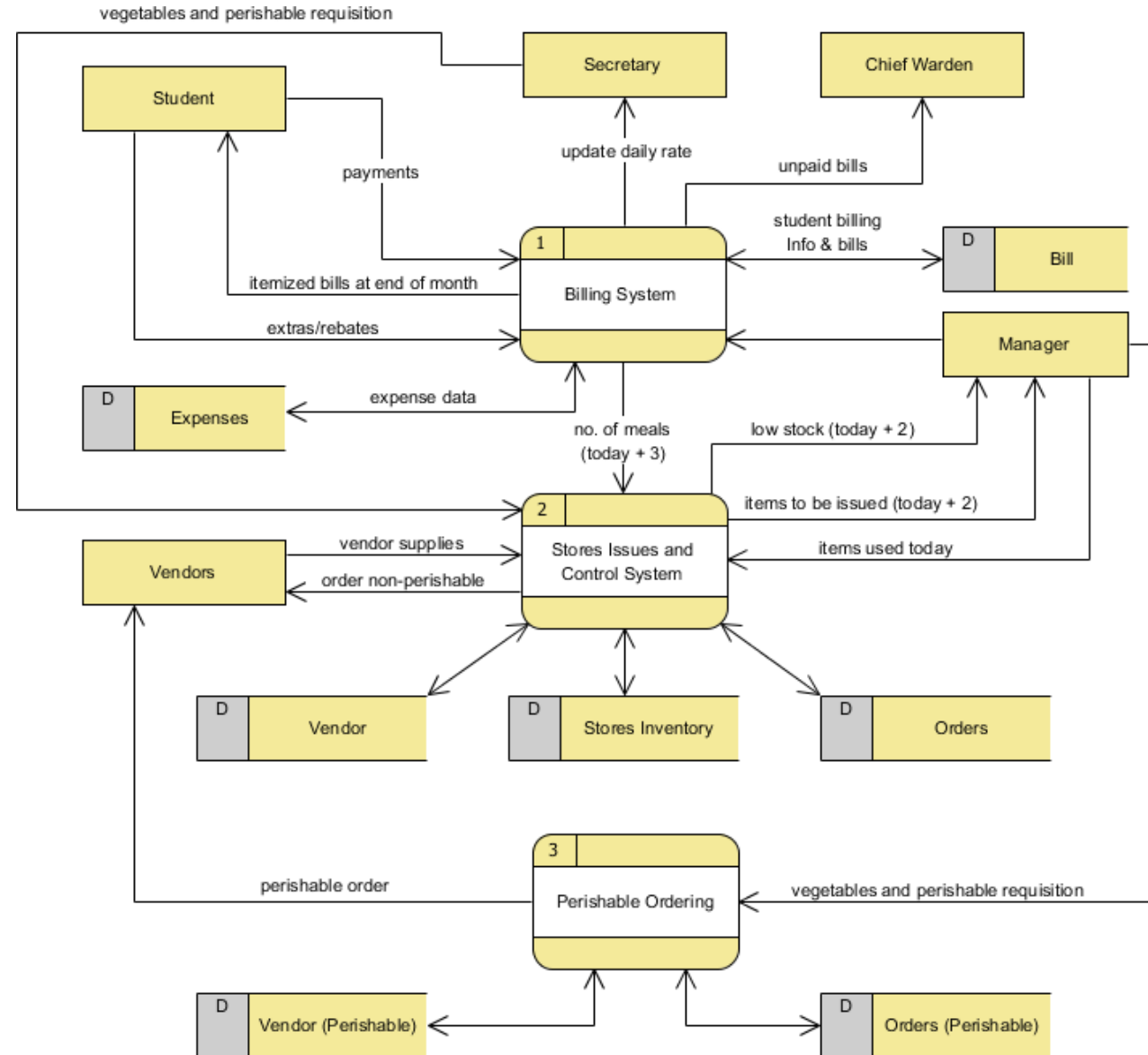
Use Case Diagram for ATM

Data Flow Diagrams (DFD)

- Represents how data moves through a system.
- **Components:**
  - **Processes** (Circle)
  - **Data Stores** (Rectangle)
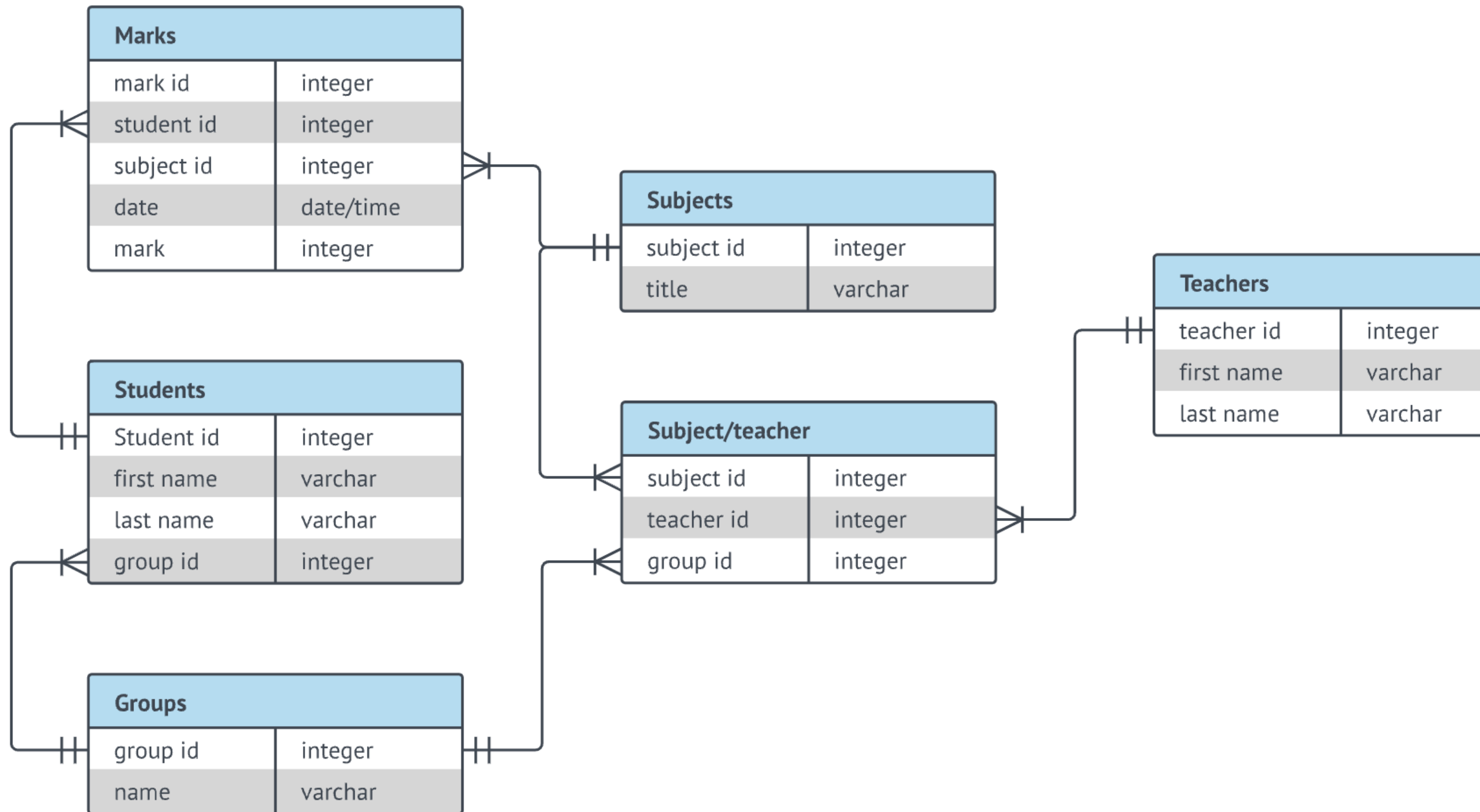 - **External Entities** (Square)
  - **Data Flows** (Arrows)

Entity-Relationship Diagram (ERD)

- **Definition**: Diagram that models database structure.

- **Key Elements**:
  - Entities (Objects like Students, Courses)
  - Attributes (Properties like Student Name, Course Code)
  - Relationships (Associations like Enrolled-In)

Nilai UNIVERSITY

Marks

mark id	integer
student id	integer
subject id	integer
date	date/time
mark	integer

Subjects

subject id	integer
title	varchar

Teachers

teacher id	integer
first name	varchar
last name	varchar

Students

Student id	integer
first name	varchar
last name	varchar
group id	integer

Subject/teacher

subject id	integer
teacher id	integer
group id	integer

Groups

group id	integer
name	varchar

Analysis for Understanding the Domain and Requirements

- **Domain Analysis:**
  - Identifying system environment, stakeholders, and constraints.
  - Techniques: **Interviews, Questionnaires, Document Analysis.**

- **Requirement Analysis Approaches:**
  - Structured Analysis (DFD, ERD)
  - Object-Oriented Analysis (UML Diagrams, Class Diagrams)

Use Case Description Format

- **Use Case Name**: Login Process
- **Actors**: User, Authentication System
- **Preconditions**: User must have an account
- **Main Flow**:
  - User enters username and password.
  - System verifies credentials.
  - User is granted access.
- **Alternative Flow**: Wrong password → Show error message.

Formal Specification

- **Definition**: Mathematically-based techniques to specify software requirements.

- **Why is it useful?**
  - Removes ambiguity.
  - Improves verification and validation.

- **Common Methods:**
  - **Z-Notation, B-Method, VDM (Vienna Development Method)**

Benefits of Using Formal Specification

- Precise and unambiguous.

- Reduces misinterpretation.

- Helps with system validation before development.

Challenges in Requirements Analysis

- **Common Issues:**
  - Unclear or vague requirements.
  - Changing requirements.
  - Conflicting stakeholder interests.
- **How to overcome them?**
  - Regular communication with stakeholders.
  - Using prototypes and visual models.

Real-World Example – Software Requirement Failure

- **Case Study**: London Ambulance System Failure
  - System designed without clear requirements.
  - Lack of proper analysis led to incorrect dispatching.
  - Lesson: Proper requirement engineering prevents system failures.

Industry Best Practices in Requirements Engineering

- **Define SMART Requirements:**
  - Specific, Measurable, Achievable, Relevant, Time-bound.

- **Use Requirement Management Tools:**
  - JIRA, IBM Rational DOORS, ReqSuite.

Summary

- **Requirement Analysis** ensures clear, complete, and correct requirements.

- **Modeling Techniques** help visualize requirements for better understanding.

- **Use Case Modeling** captures user interactions.

- **Formal Specification** provides a rigorous, mathematical approach to defining requirements.

End of Topic 3

Nilai
UNIVERSITY