Analysis Modelling

Slide Set - 6

Organized & Presented By:
Software Engineering Team CSED
TIET, Patiala

Why Analysis Modeling ?

- Provides the first technical representation of a system
- Is easy to understand and maintain
- Deals with the problem of size by partitioning the system
- Uses graphics whenever possible
- Differentiates between essential information versus implementation information
- Helps in the tracking and evaluation of interfaces
- Provides tools other than narrative text to describe software logic and policy

Requirements Analysis

Purpose

- Specifies the software's operational characteristics
- Indicates the software's interfaces with other system elements
- Establishes constraints that the software must meet
- Provides the software designer with a representation of information, function, and behavior
  - This is later translated into architectural, interface, class/data and component-level designs
- Provides the developer and customer with the means to assess quality once the software is built

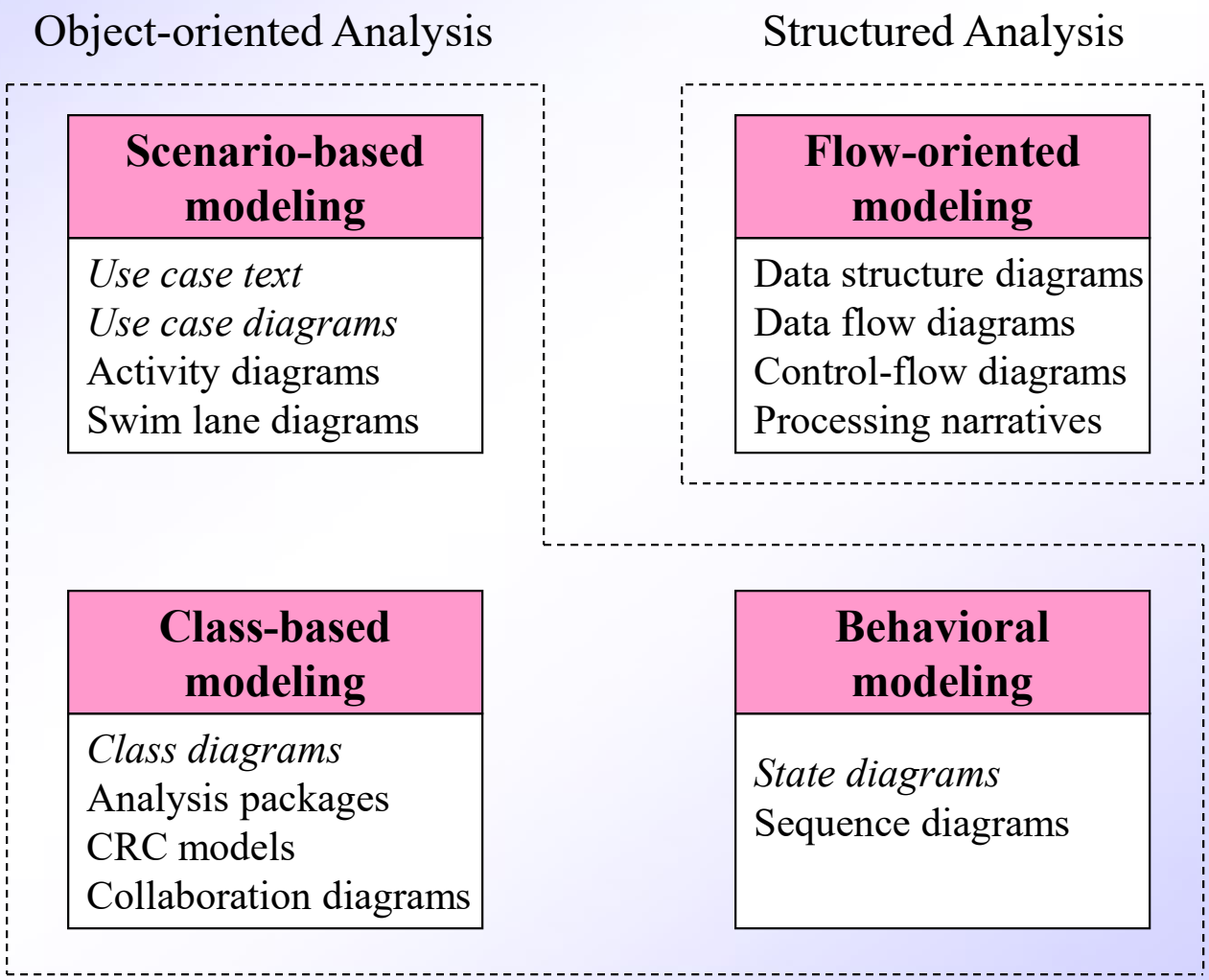
Who Carries Out Requirements Analysis and Specification?

- The person who undertakes requirements analysis and specification:
  - Known as systems analyst:
  - Collects data pertaining to the product
  - Analyzes collected data:
    - To understand what exactly needs to be done.
  - Writes the Software Requirements Specification (SRS) document.

A Set of Models

- **Flow-oriented modeling** – provides an indication of how data objects are transformed by a set of processing functions
- **Scenario-based modeling** – represents the system from the user's point of view
- **Class-based modeling** – defines objects, attributes, and relationships
- **Behavioral modeling** – depicts the states of the classes and the impact of events on these states

Elements of the Analysis Model



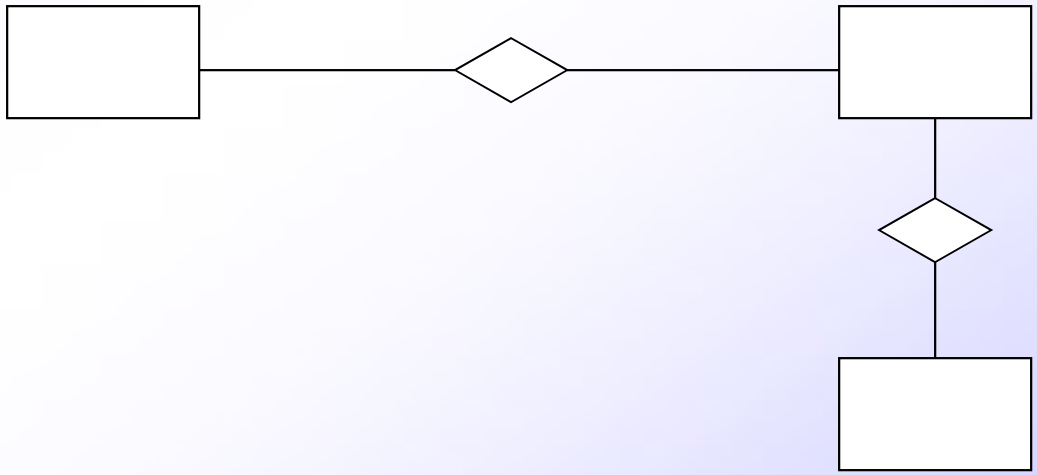
Analysis Modeling Approaches

- Structured analysis
  - Considers data and the processes that transform the data as separate entities
  - Data is modeled in terms of only attributes and relationships (but no operations)
  - Processes are modeled to show the 1) input data, 2) the transformation that occurs on that data, and 3) the resulting output data
- Object-oriented analysis
  - Focuses on the definition of classes and the manner in which they collaborate with one another to fulfill customer requirements

Flow-oriented Modeling

Data Modeling

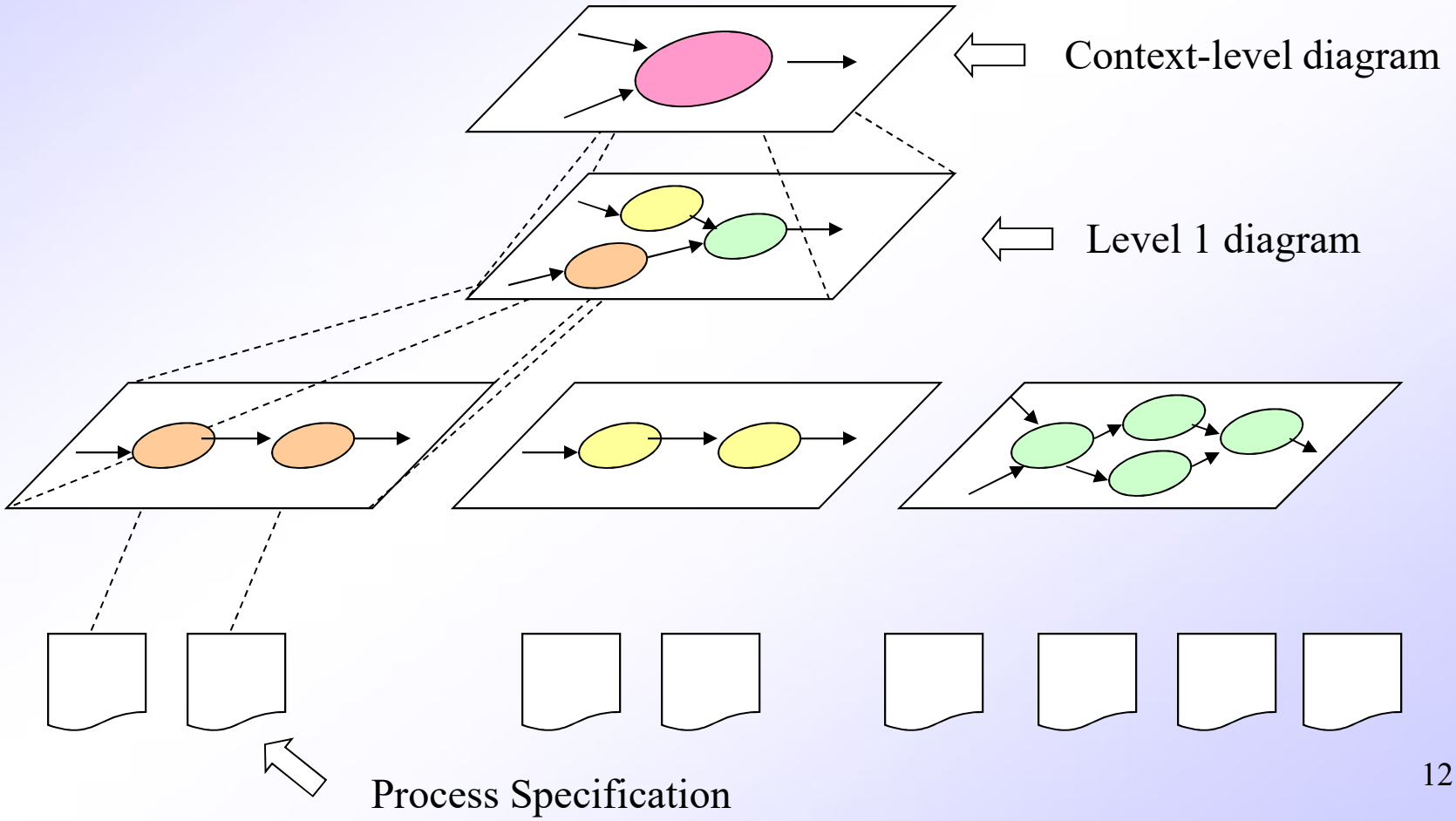
- Identify the following items
  - Data objects (Entities)
  - Data attributes
  - Relationships
  - Cardinality (number of occurrences)



Data Flow and Control Flow

- Data Flow Diagram
  - Depicts how input is transformed into output as data objects move through a system
- Process Specification
  - Describes data flow processing at the lowest level of refinement in the data flow diagrams
- Control Flow Diagram
  - Illustrates how events affect the behavior of a system through the use of state diagrams

Diagram Layering and Process Refinement



Scenario-based Modeling

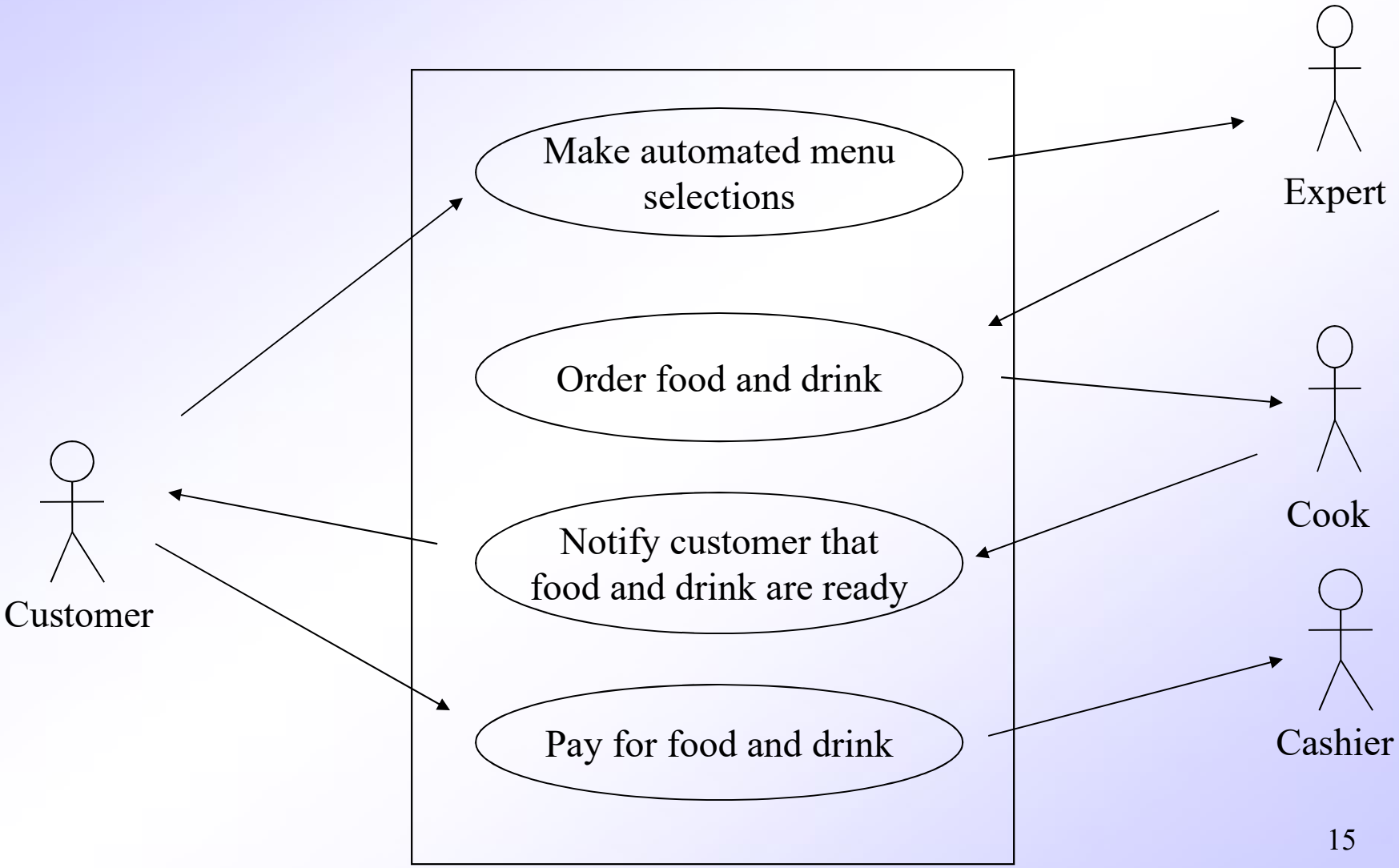
Writing Use Cases

- It is effective to use the first person "I" to describe how the actor interacts with the software
- Format of the text part of a use case

Use-case title: Actor: Description: I …

(See examples in Pressman textbook on pp. 188-189)

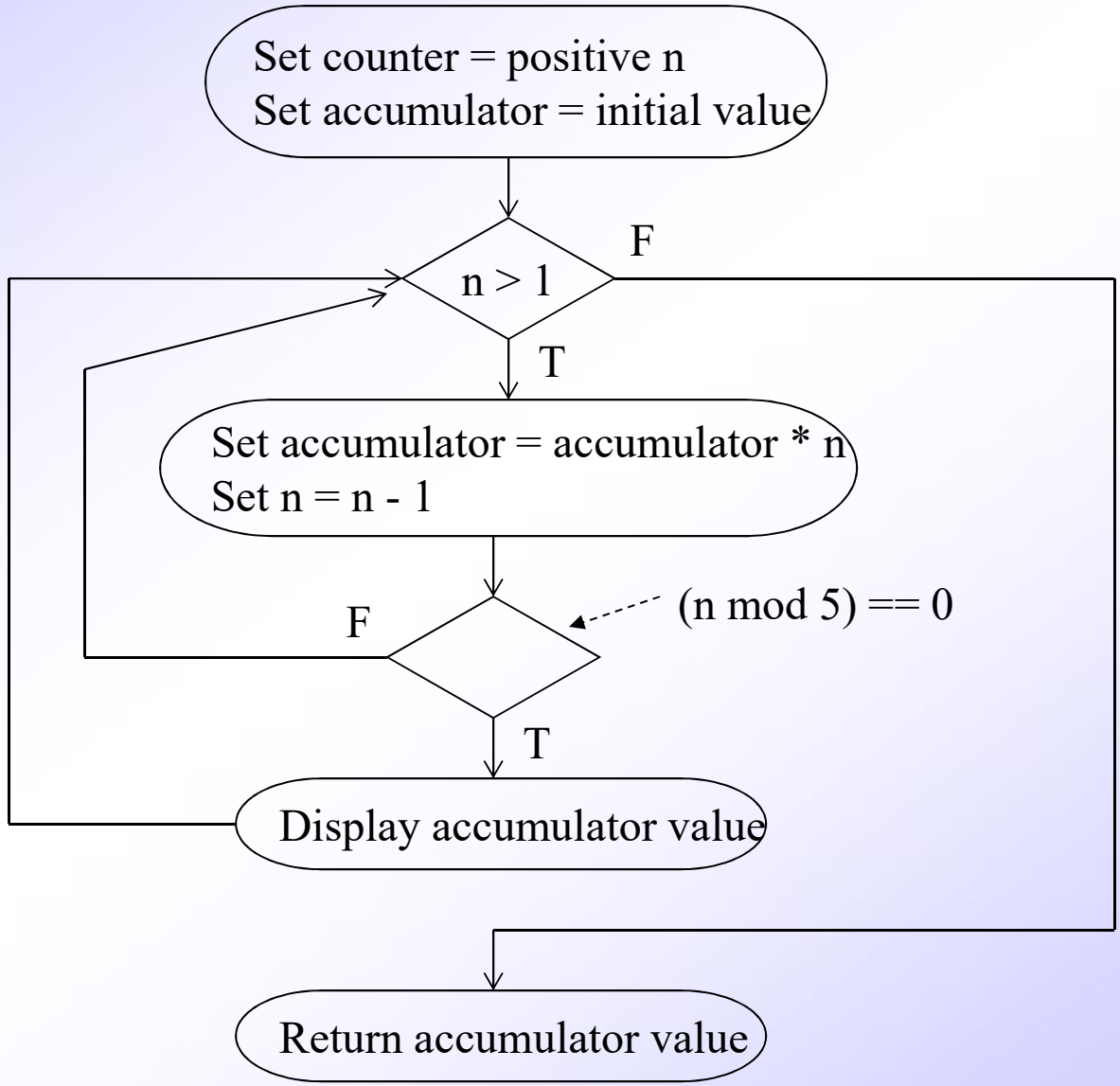
Example Use Case Diagram



Activity Diagrams

- Supplements the use case by providing a graphical representation of the flow of interaction within a specific scenario
- Uses flowchart-like symbols
  - **Rounded rectangle** - represent a specific system function/action
  - **Arrow** - represents the flow of control from one function/action to another
  - **Diamond** - represents a branching decision
  - **Solid bar** – represents the fork and join of parallel activities

Example Activity Diagram



Class-based Modeling

Identifying Analysis Classes

1) Classes are determined by underlining each noun or noun clause
2) A class should NOT have an imperative procedural name (i.e., a verb)

- General classifications for a class
  - External entity (e.g., another system, a device, a person)
  - Thing (e.g., report, screen display)
  - Occurrence or event (e.g., movement, completion)
  - Role (e.g., manager, engineer, salesperson)
  - Organizational unit (e.g., division, group, team)
  - Place (e.g., manufacturing floor, loading dock)
  - Structure (e.g., sensor, vehicle, computer)

(More on next slide)

Defining Attributes of a Class

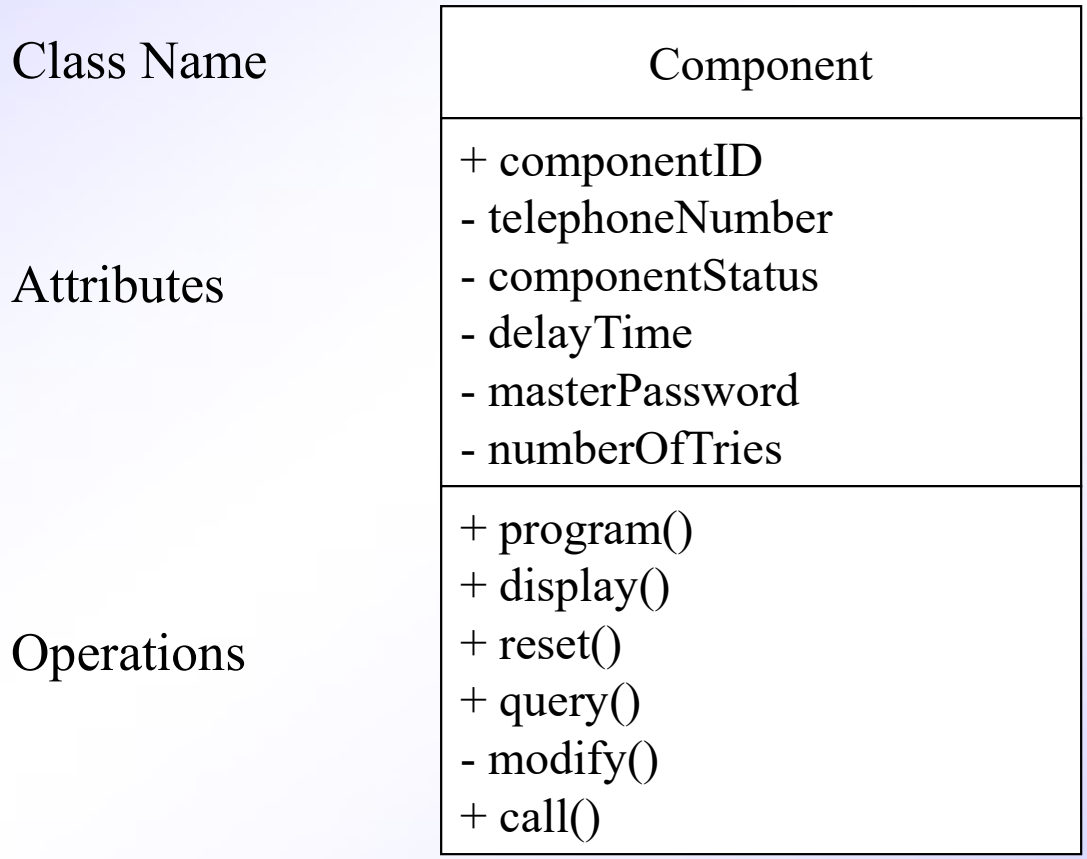
- Attributes of a class are those nouns from the grammatical parse that reasonably belong to a class
- Attributes hold the values that describe the current properties or state of a class

Defining Operations of a Class

- Operations define the behavior of an object

- Four categories of operations
  - Operations that manipulate data in some way to change the state of an object (e.g., add, delete, modify)
  - Operations that perform a computation
  - Operations that inquire about the state of an object
  - Operations that monitor an object for the occurrence of a controlling event

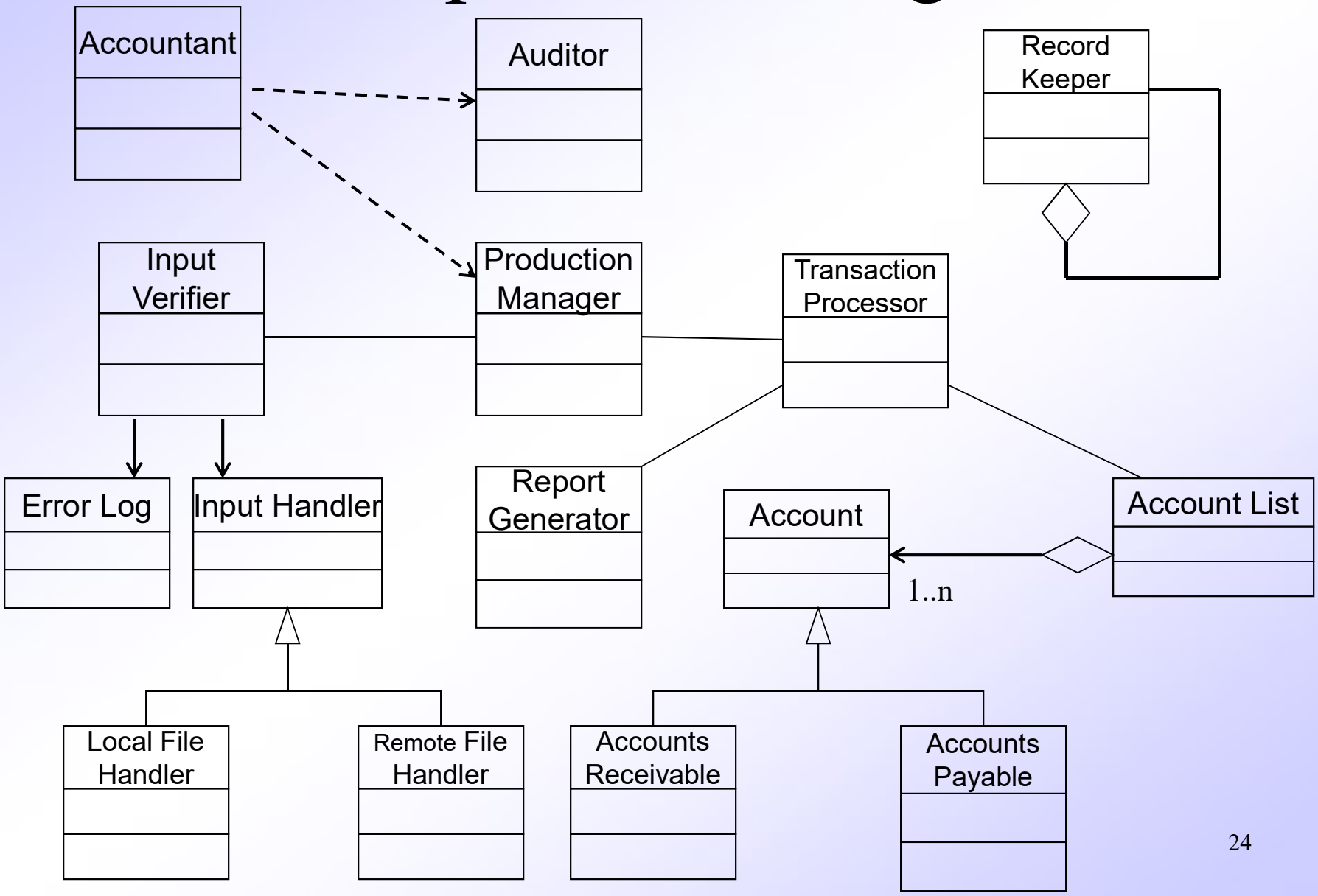
Example Class Box



Association, Generalization and Dependency (Ref: Fowler)

- Association
  - Represented by a solid line between two classes directed from the source class to the target class
  - Used for representing (i.e., pointing to) object types for attributes
  - May also be a part-of relationship (i.e., aggregation), which is represented by a diamond-arrow
- Generalization
  - Portrays inheritance between a super class and a subclass
  - Is represented by a line with a triangle at the target end
- Dependency
  - A dependency exists between two elements if changes to the definition of one element (i.e., the source or supplier) may cause changes to the other element (i.e., the client)
  - Examples
    - One class calls a method of another class
    - One class utilizes another class as a parameter of a method

Example Class Diagram



Behavioral Modeling

Creating a Behavioral Model

1) Identify events found within the use cases and implied by the attributes in the class diagrams

2) Build a state diagram for each class, and if useful, for the whole software system

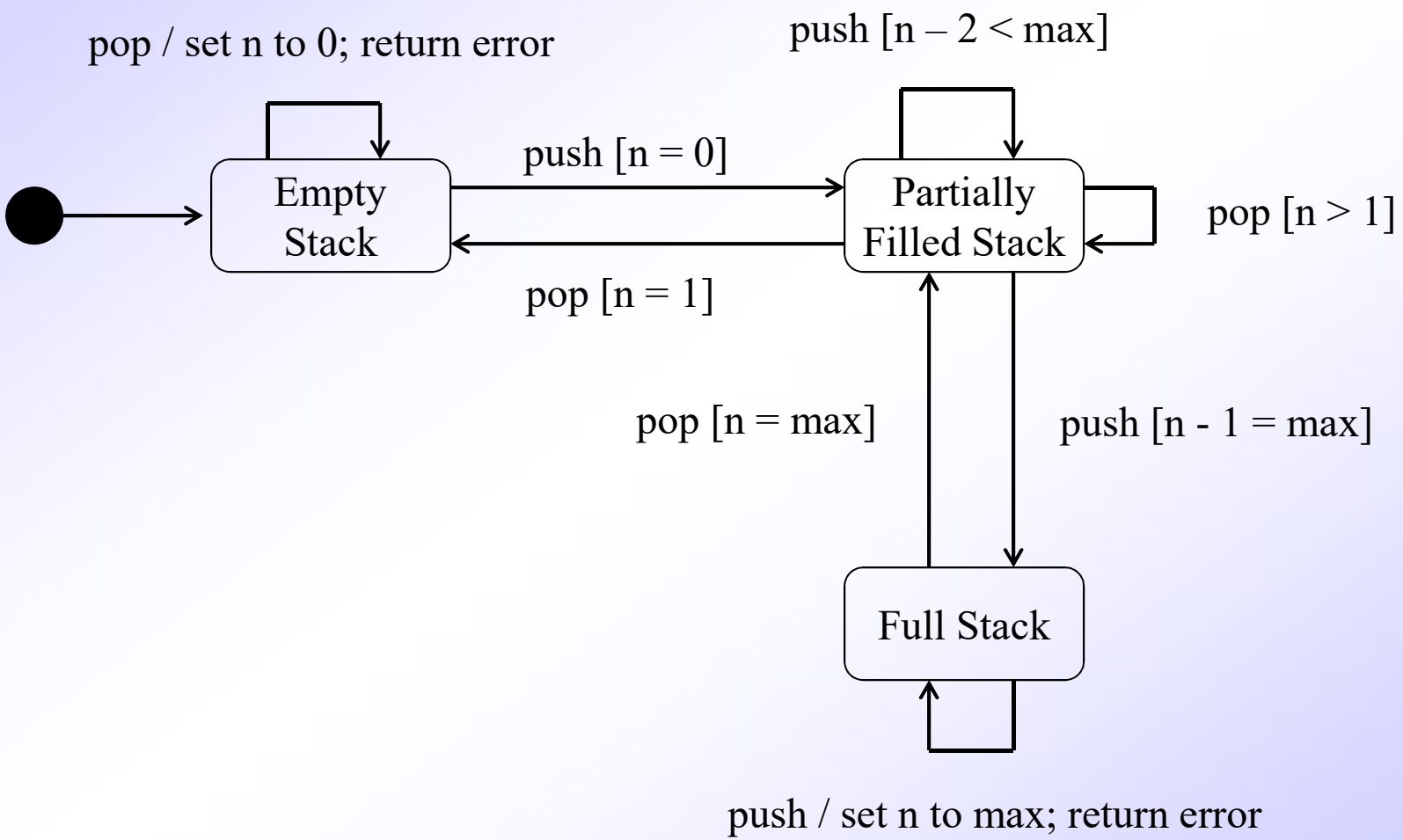
Identifying Events in Use Cases

- An event occurs whenever an actor and the system exchange information
- Some events have an explicit impact on the flow of control, while others do not

Building a State Diagram

- A state is represented by a rounded rectangle
- A transition (i.e., event) is represented by a labeled arrow leading from one state to another
  - `Syntax: trigger-signature [guard]/activity`

Example State Diagram



Summary: Elements of the Analysis Model