POAD Book:

# Chapter 8

# POAD: Analysis Phase

Instructor: Dr. Hany H. Ammar

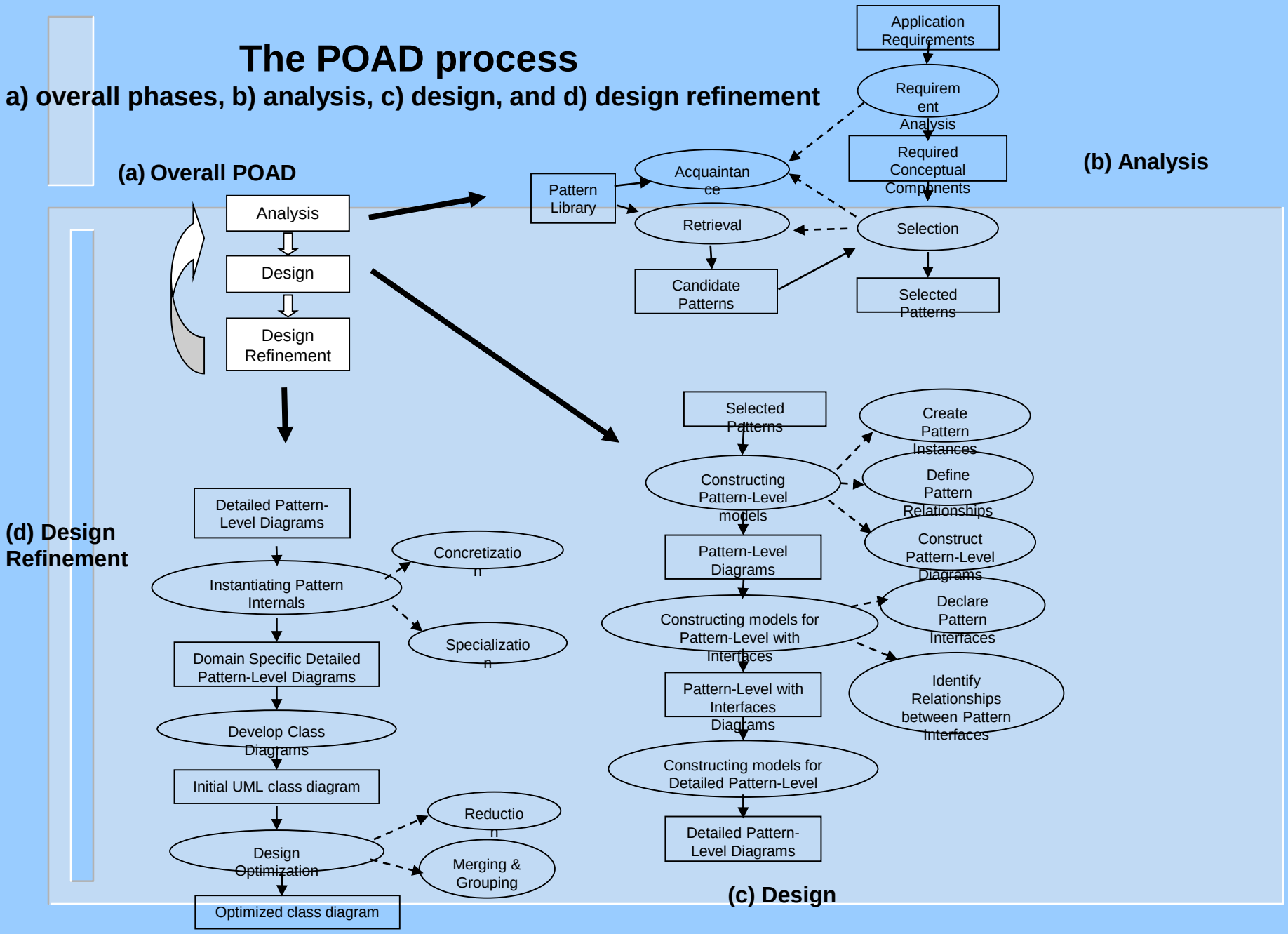
Dept. of Computer Science and Electrical Engineering, WVU

# Items to be covered

- Review of the POAD Process
- The analysis phase process
- Requirement Analysis
- Acquaintance with relevant pattern databases
- Retrieval of patterns from the domain specific databases
- Selection of patterns from a set of candidate patterns
- Example case studies

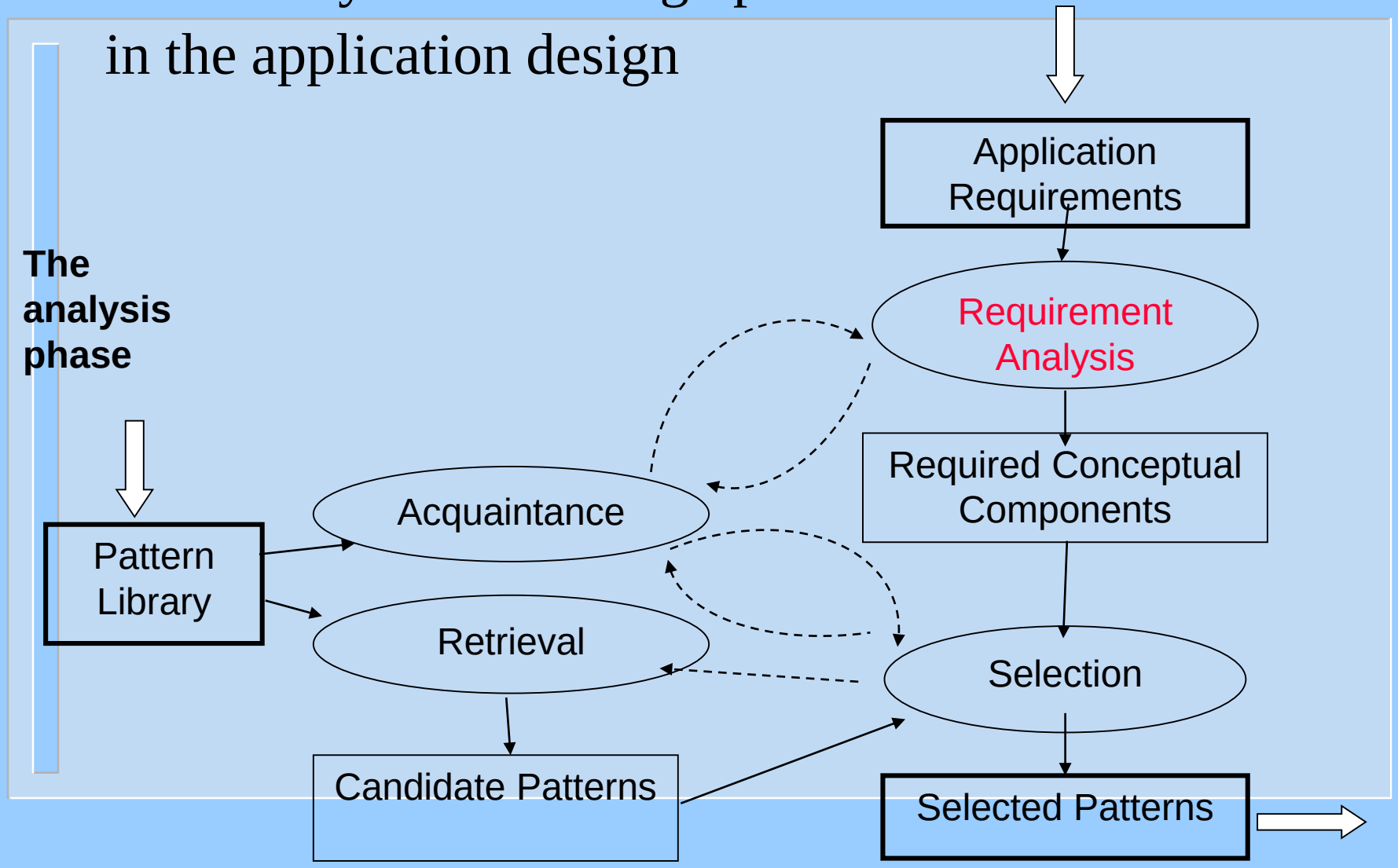
# The POAD process

## a) overall phases, b) analysis, c) design, and d) design refinement



# Purpose

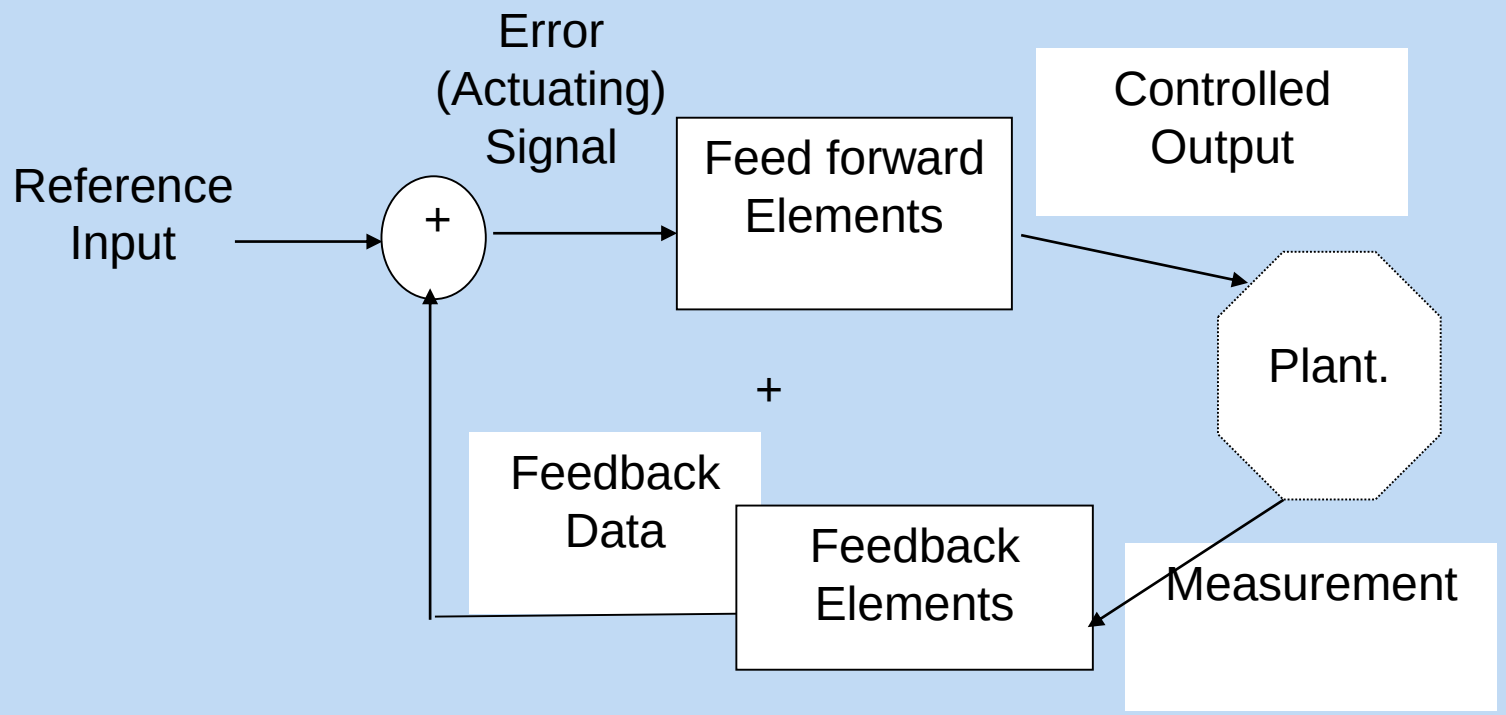
- To identify a set of design patterns that will be used in the application design



# Requirements Analysis

- Purpose
  - To analyze the application requirements and identify the possible problems to be solved
  - To determine conceptual components
- Process
  - Finding Components
    - Identify Functionality; articulate problems to be addressed.
    - Generate use cases

# Example: The Feedback Control System



# Example: The Feedback Control System

- To implement a feedback control system, the specification and description of the system configuration and its components must be put into a form amenable for analysis and design.
- The portion of a system to be controlled is usually called the *Plant*
- The system is decomposed into the following components

# Example: The Feedback Control System

- *feedforward* component - processes error data and applies control algorithm to the plant.
- *feedback* component - measures data from plant, processes it, and provides feedback data.
- *error calculation* component - compares input and feedback data and produces error.
- *plant* - external component on which control is applied and from which measurements are taken

# Requirements Analysis Process (cont'd)

- Finding Components (cont'd)
- Feedback control system required to regulate quality in a production line
  - We identify the use case Observe Changes in plant
  - How do we do that
    - This brings us to UML diagrams
- Assigning Responsibilities to Components
  - Determine which components will fulfill which functionality
    - Observing changes in plant
    - Feeding the information back

# Requirements Analysis

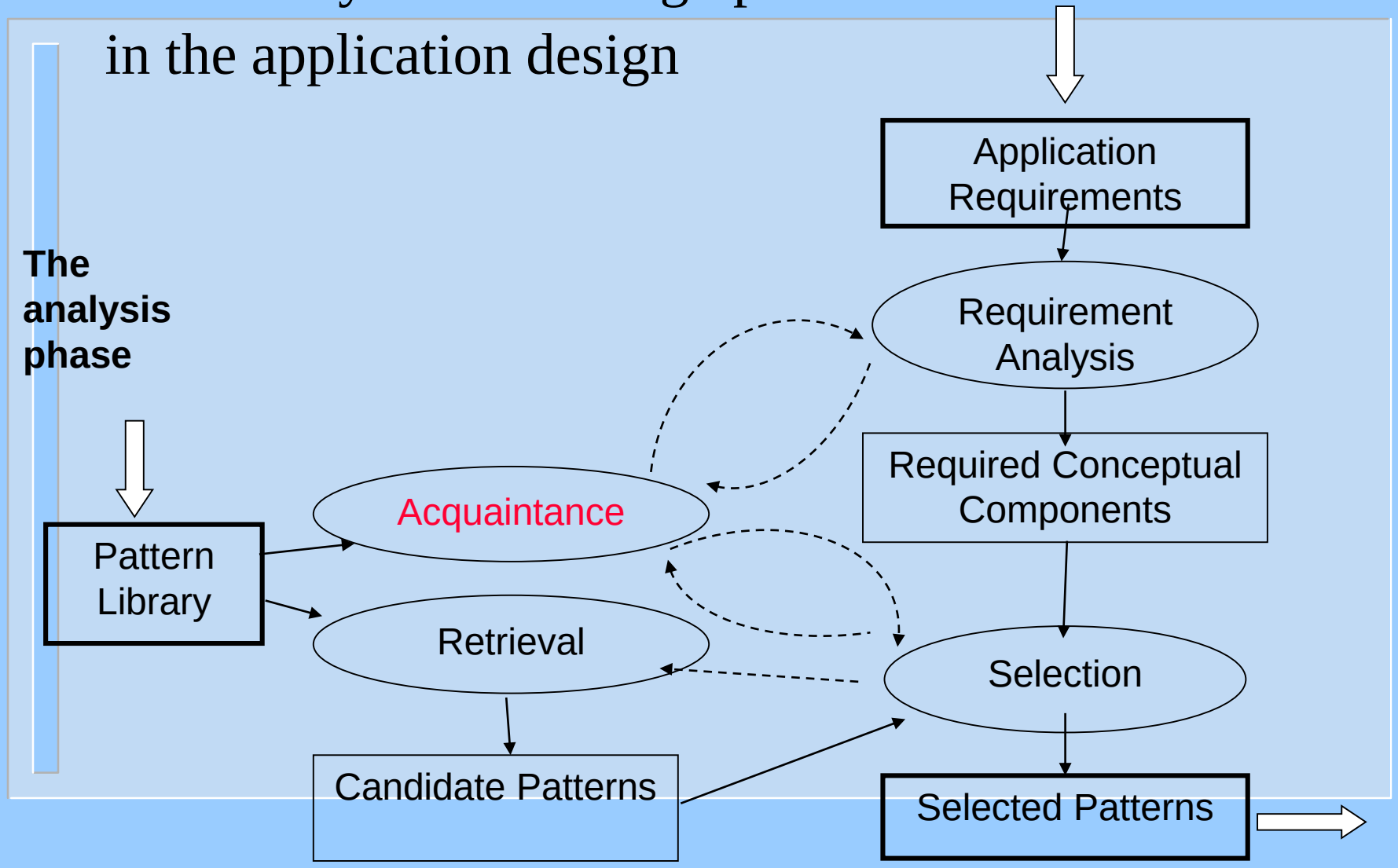
- Process (cont'd)
  - Relationship to S/W architecture
    - Notice in any S/W development Methodology, we want to identify the application components
    - There is one Difference for the components in POAD
      - S/W architectural components are deliverable units
      - POAD components or conceptual are abstract (logical-level)
  - Analyzing Large Applications
    - Large Systems are broken down into subsystems
    - Subsystems are analyzed in more details

# Requirements Analysis

- Iterating With Other Analysis Activities
  - Acquaintance Activity- Look for ideas from previously developed systems, documented solutions in libraries or databases
  - Requirements Analysis relies on Acquaintance
- Product
  - Set of conceptual components, use cases and responsibilities (UML Requirements models and UML analysis models)

# Purpose

- To identify a set of design patterns that will be used in the application design



# Acquaintance

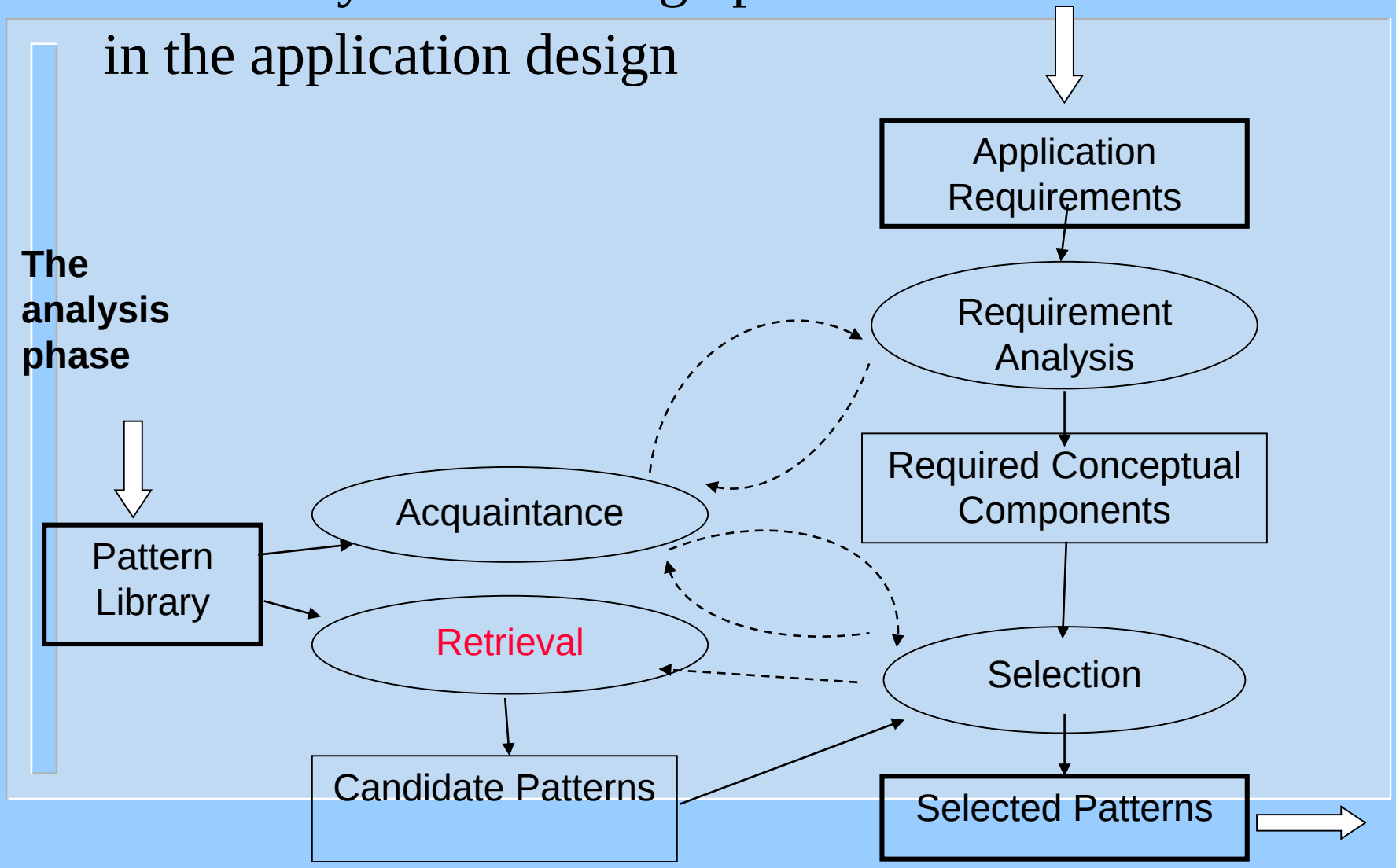
- Purpose- To get analyst familiar to existing solutions in application domain
- Process
  - Pattern Database Activities
    - Determine Which components can be isolated into patterns from the database
    - Identify fragments of existing solutions to build overall solution

# Acquaintance

- Rationale
  - Attract analyst towards existing good solutions
  - Some existing patterns may motivate the analyst to identify conceptual components (e.g. Agent-based systems)
- Iterations
  - There will be several iterations between acquaintance activity and requirements analysis
  - There will be iterations also between the selection activity and acquaintance.
- Product – Knowledge gained about existing solutions

# Purpose

- To identify a set of design patterns that will be used in the application design



# Retrieval

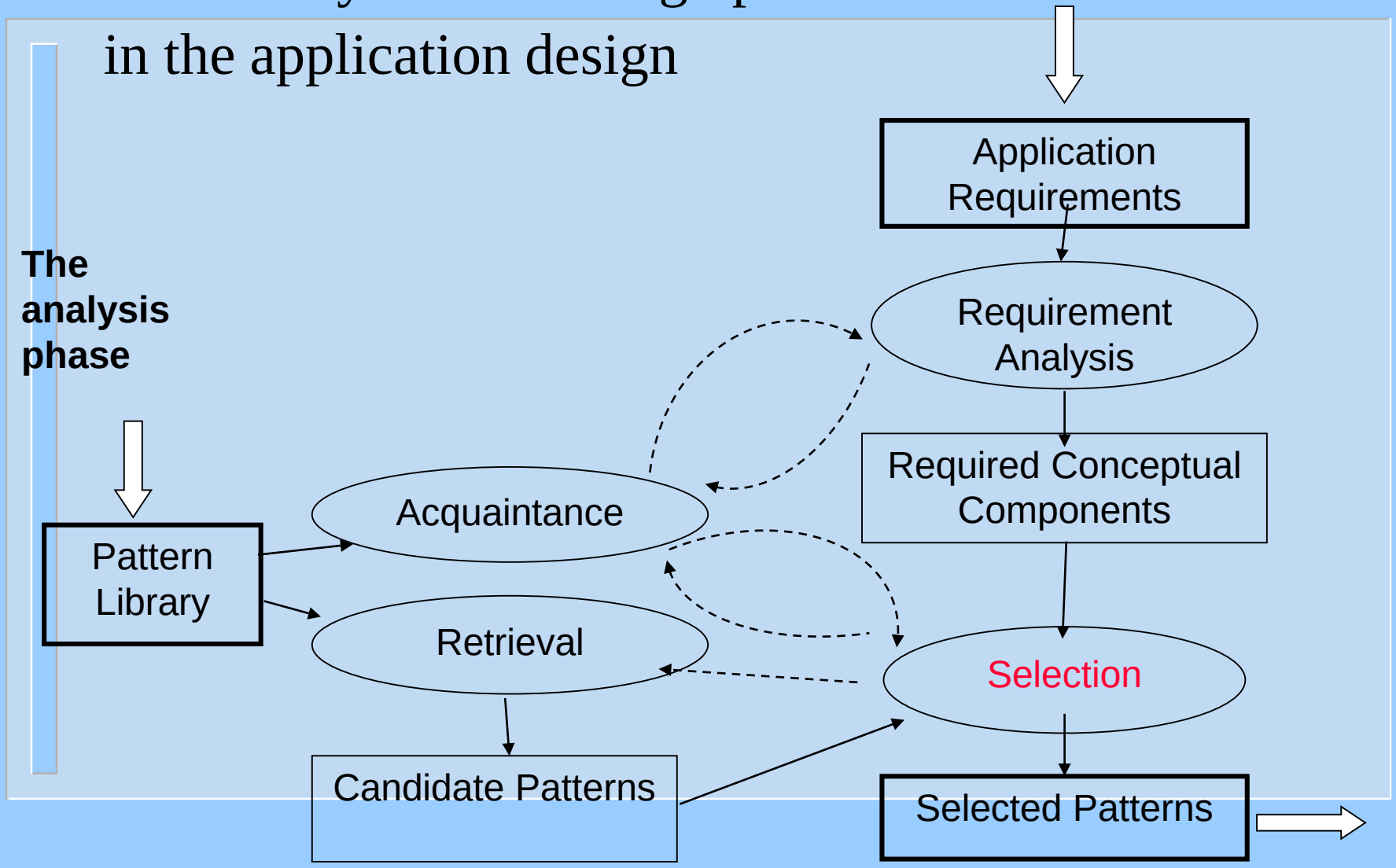
- Purpose- Retrieve set of relevant patterns from database.
- Process
  - Focuses on how to select a given pattern
  - The Database Query
    - Retrieval depends on matching candidate Asset to Database Query
  - Navigation
    - Defines how the assets in the library are visited

# Retrieval

- Matching
  - Testing relevance of an asset and deciding if it should be selected
- Retrieval goals- Every Query must have a goal
  - E.g., retrieve all patterns that have an observer or monitoring strategy
  - Then assess the precision and recall of the techniques.
- Product- Set of candidate Design Patterns

# Purpose

- To identify a set of design patterns that will be used in the application design



# Pattern Selection

- Purpose- Select a set of patterns that fulfills the responsibilities of each component
  - Process
    - Problem Solution Matching
      - Set of Components define Problems
      - Solutions are defined by set of candidate patterns
    - Difficulty in Selection
      - Human Intensive
      - Maturity- Methodologies that use patterns aren't mature as other ones

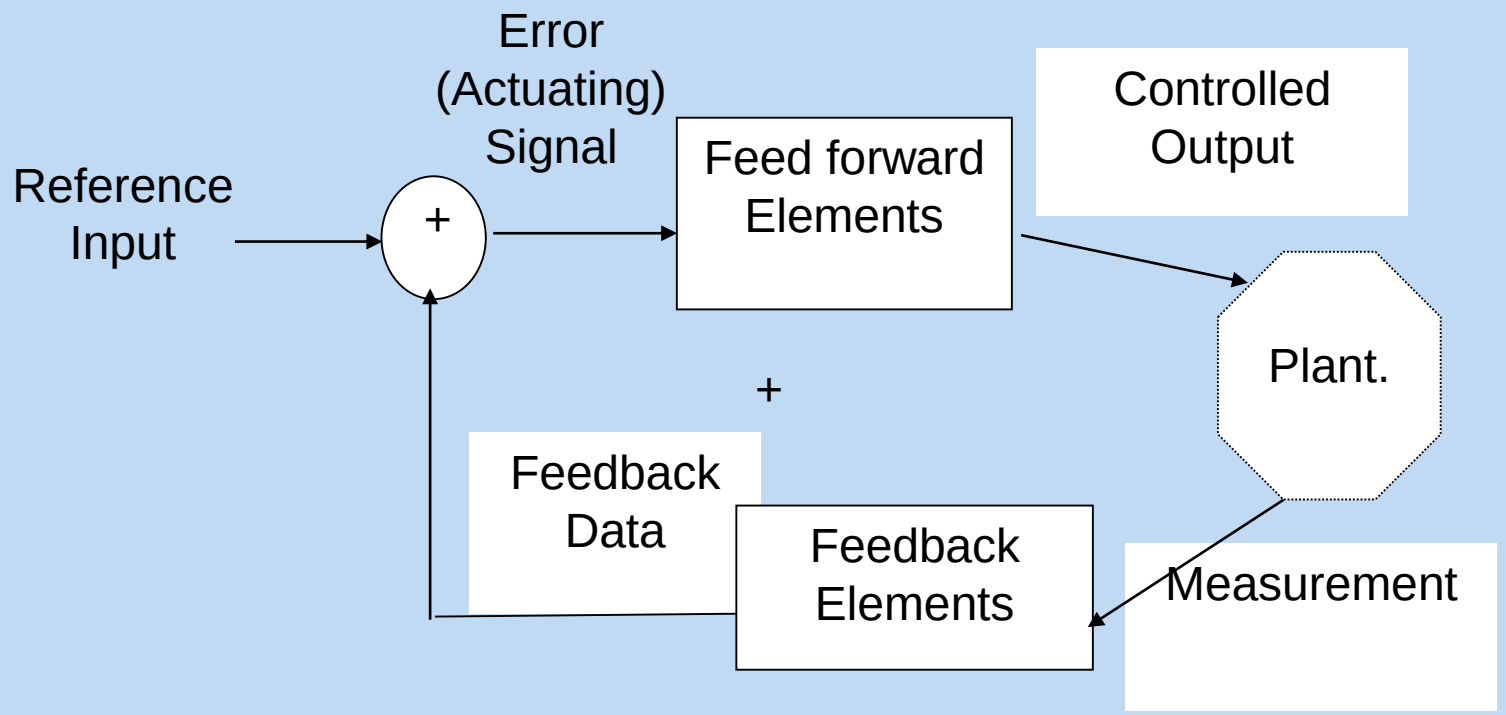
# Pattern Selection

- Iteration- Analyst revisits the acquaintance and retrieval activities
- Domain Specific Libraries – Contain patterns related to the specific problem, leads to easier acquaintance.
- Study Relationships between patterns, Know which patterns:
  - Use other patterns
  - Refines other patterns
  - Conflicts with another pattern
- Product – Set of patterns used in designing the application

# Items to be covered

- Review of the POAD Process
- The analysis phase process
- Requirement Analysis
- Acquaintance with relevant pattern databases
- Retrieval of patterns from the domain specific databases
- Selection of patterns from a set of candidate patterns
- Example case studies

# Example 1: The Feedback Control System



# Example: The Feedback Control System

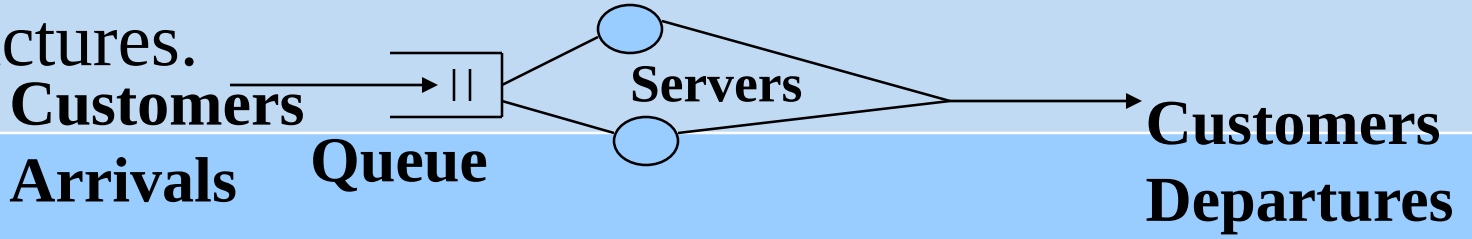
- The *feedforward* component implements some sort of a control strategy- *strategy* pattern is a good candidate for this task
- *feedback* component receives measurements & applies  feedback control strategy
  - measurement observations are communicated to feedback controller using the *Observer* pattern
  - measured data is fed to feedback control strategy, which should provide flexibility to plug in and plug out different feedback control strategies.

# Example: The Feedback Control System

- This can be implemented using another *Strategy* pattern
- the *error calculation* - feedback controller notifies the error calculation unit with the feedback data.
  - *Observer* pattern can implement this behavior
- *Blackboard* pattern for managing the system repository of measured data, feedback data and error data

# Example 2: Chapter 12 Simulation of waiting Queues

- Developing a domain-specific design for the purpose of producing applications that simulate the behavior of waiting queues.
- Customers lining up for service from one or more service stations (or in general, requests or transactions queue up for service from servers)
- The purpose of developing such simulation environments is to be able to get some measurements about systems that implement waiting-queues structures.



# Example 2: Chapter 12 Simulation of waiting Queues

- There are several degrees of variances that should be accommodated in the design:
  - *Topology of service stations:* The number of service stations may be fixed or variable (in time).
  - *Service time*: The service time may be constant or variable.
  - *Topology of queues:* We may have a single queue, multiple interchangeable queues, or multiple queues with different service categories

# Example 2: Chapter 12 Simulation of waiting Queues

- *Types of queues:* Queues could be first in, first out (FIFO) queues; last in, first out (LIFO) queues (stacks); priority queues; or limited size queues.
- *Arrival distribution:* Customers arrive to the system at various rates. The distribution of the customer arrivals in time could be Markovian distribution, Poisson distribution,
- *Dispatching policy: how c*ustomers are assigned to queues ? (e.g. random)
- *Measurements:* As a result of the simulation, measure the customers waiting time, and throughput

# Example 2: Chapter 12 Simulation of waiting Queues

- Requirements Analysis: The following components are identified
  - *Customer generator*.The customer generator component uses one of a distribution function to generate the time of arrival
  - *Queue facility*.The queue facility component consists of a set of queue categories, where each queue category contains one or more queues
  - *Service facility*. The service facility component consists of a set of service categories, where each service category contains one or more servers.
  - *Event manager*.The event manager component serves as the main driver for the simulation and produces simulation measurements (results).

# Example 2: Chapter 12 Simulation of waiting Queues

- Pattern Selection
  - Customer Generator component use a distribution function to generate the next customer to arrive
    - Select the TemplateMethod pattern as an interface for generating customers (TemplateMethod pattern defines the skeleton of an algorithm (in our case study the steps required to generate a customer). The Strategy pattern is a candidate
  - The Event Manager (Scheduler) component plays the role of demultiplexing and dispatching of events
    - select the Reactor pattern to implement the design of the Event Manager (supports the “de-multiplexing and dispatching of events to multiple event handlers triggered concurrently by multiple events,

# Example 3: A Digital Content Remastering Application (POAD Ch.13)

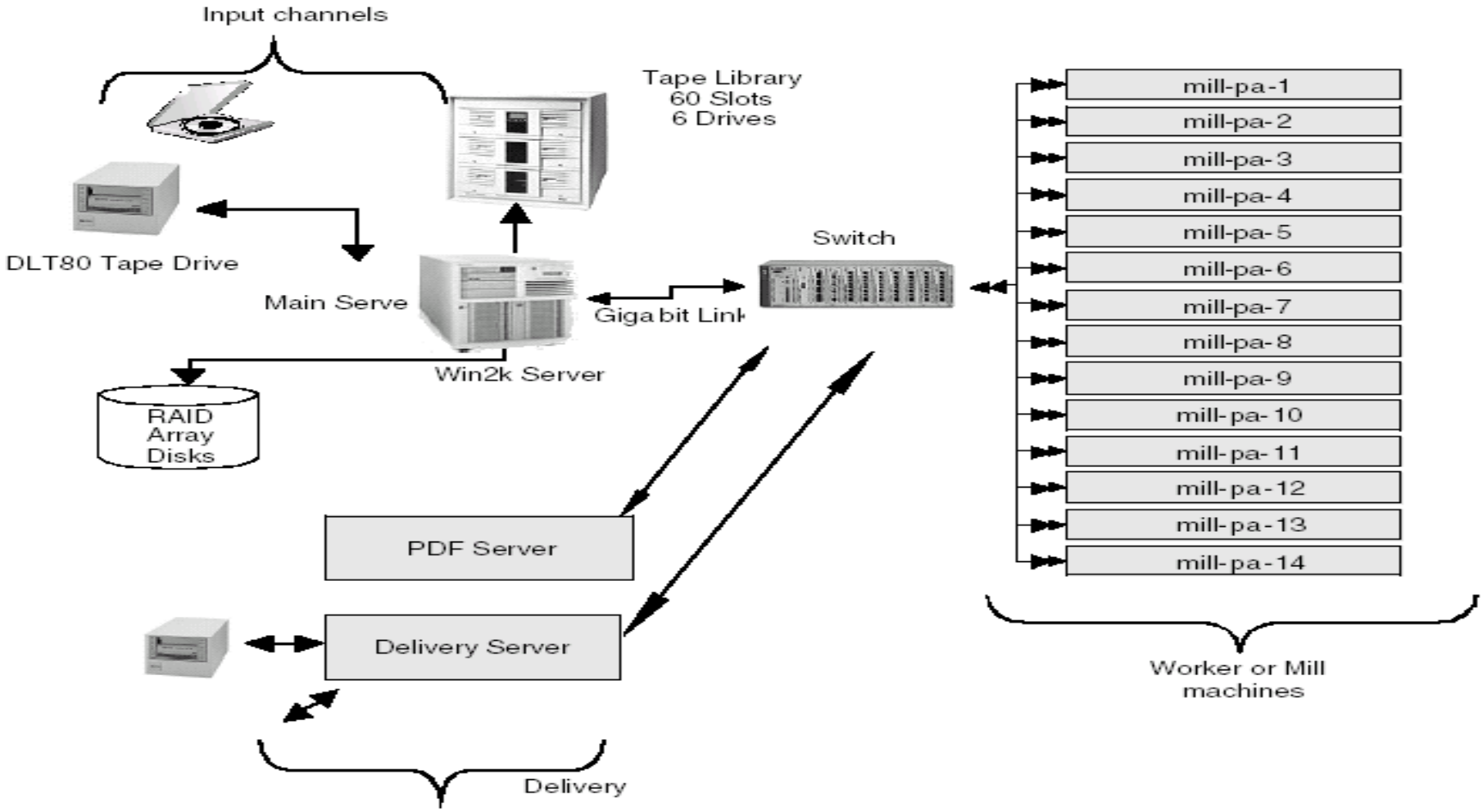



**Figure 13–1** A hardware setup for the DCRM system.

# A Digital Content Remastering (DCRM) Application (POAD Ch.13)

- Requirements: DCRM consists of several subsystems
  - The Distribution subsystem, a subsystem that deals with the distribution of tasks across the worker machines
  - The Filtering Subsystem, a subsystem that handles the application of several content-understanding algorithms
  - The controller engine subsystem that handles the workflow logic and masters the execution of components in the system.

# A Digital Content Remastering (DCRM) Application (POAD Ch.13)

- The Distribution subsystem Requirements
  - Provide the necessary communication mechanisms to communicate with all worker machines
  - Be extensible to support several communication technologies, and be able to communicate with heterogeneous worker machines
  - Provide a mechanism for distributing tasks to worker machines, and be extensible in terms of the number of machines it supports,

# A Digital Content Remastering (DCRM) Application (POAD Ch.13)

- Conceptual Components
  - Task Distribution Component
  - Communication Component
- Acquaintance, Retrieval, Pattern Selection
  - the literature is full of patterns for distributed and communication systems
  - Some candidates include the Proxy pattern and the Strategy Pattern