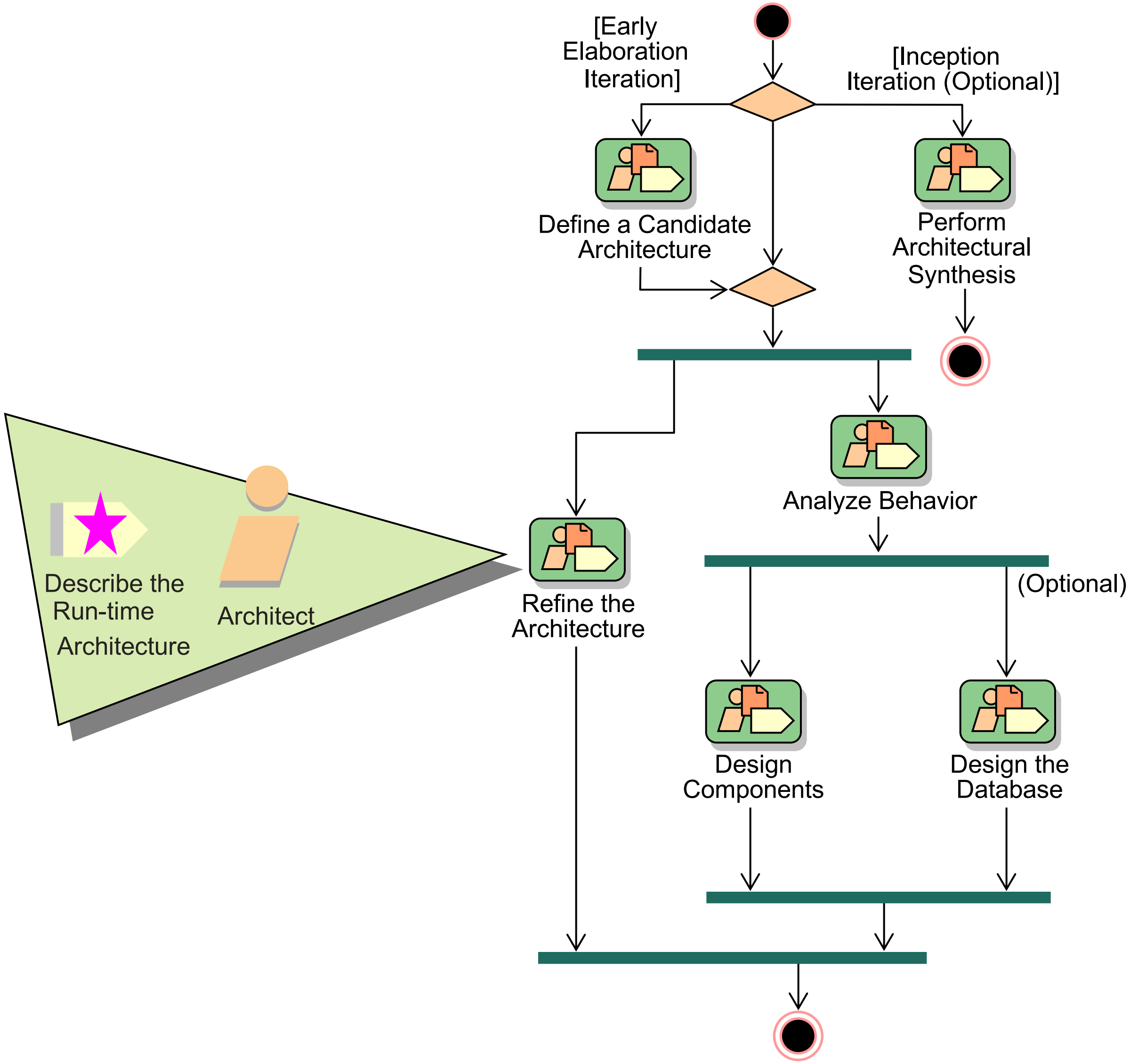
Software analysis and design

Module 13: Describe the Runtime Architecture

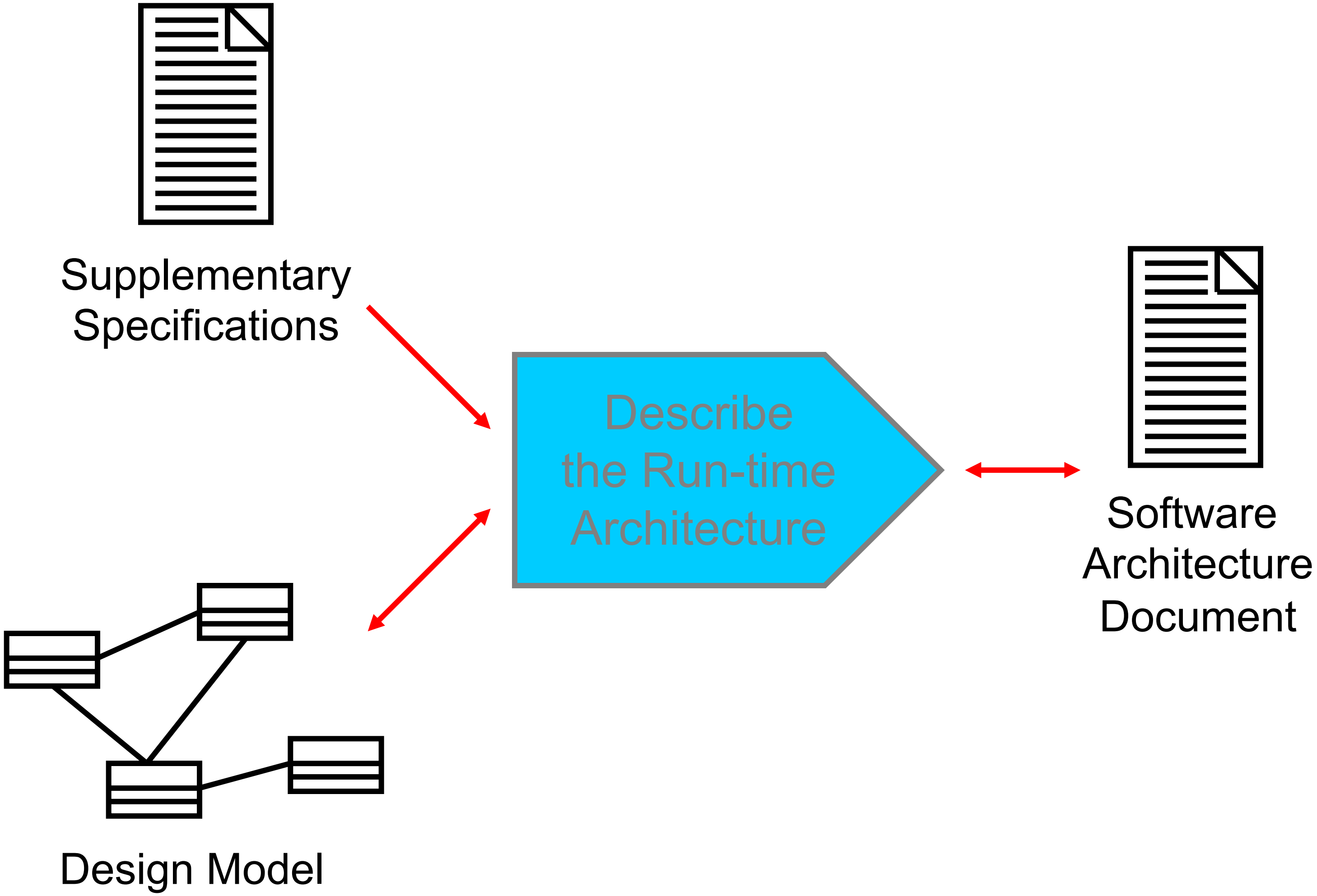
Objectives: Describe the Run-time Architecture

- Define the purpose of the Describe the Run-time Architecture activity and when in the lifecycle it is performed
- Demonstrate how to model processes and threads
- Explain how processes can be modeled using classes, objects and components
- Define the rationale and considerations that support architectural decisions

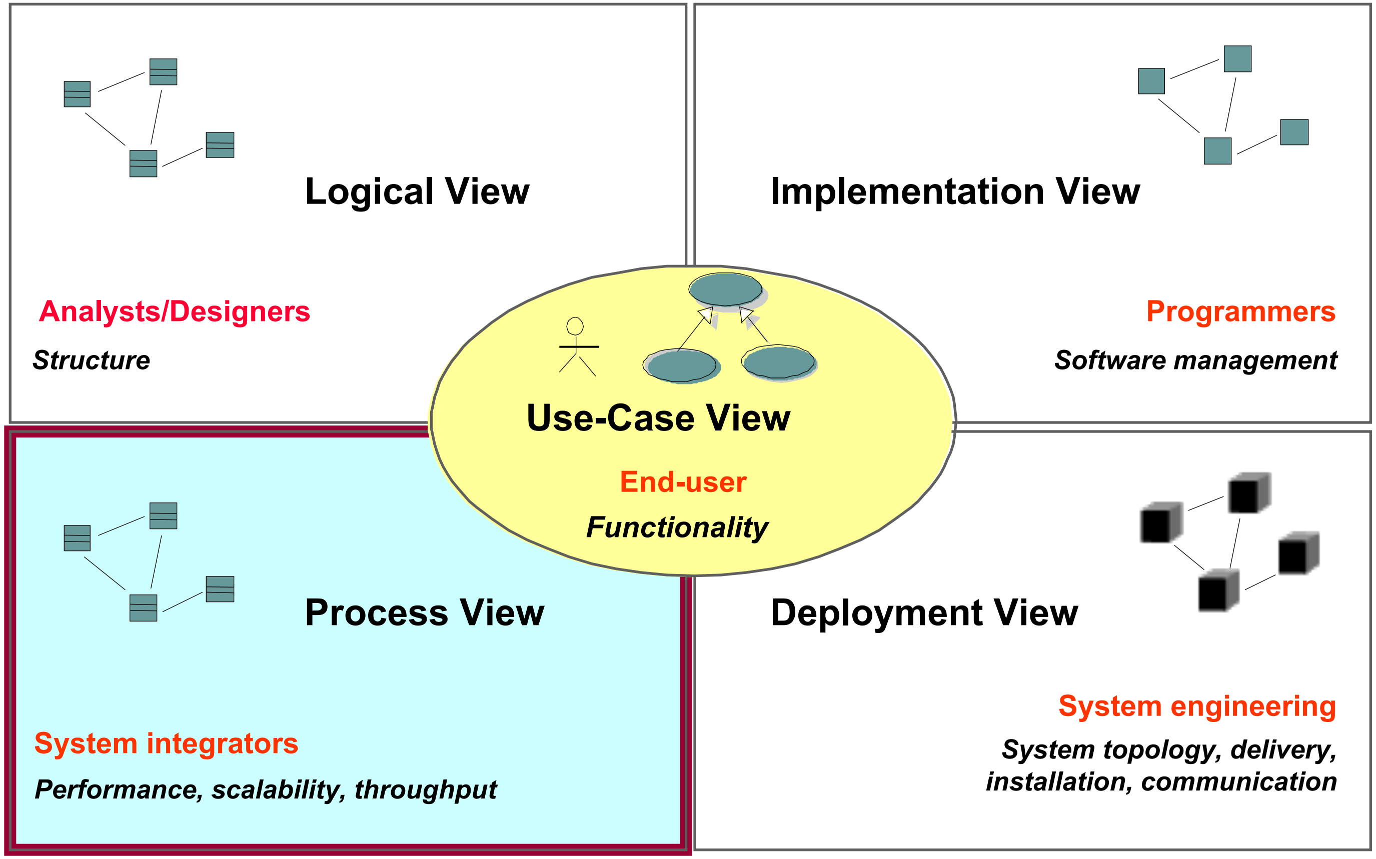
Describe the Run-time Architecture in Context


[Early Elaboration Iteration]
[Inception Iteration (Optional)]
Define a Candidate Architecture
Perform Architectural Synthesis
Analyze Behavior
Describe the Run-time Architecture
Architect
Refine the Architecture
(Optional)
Design Components
Design the Database

Describe the Run-time Architecture Overview


Supplementary Specifications
Describe the Run-time Architecture
Software Architecture Document
Design Model

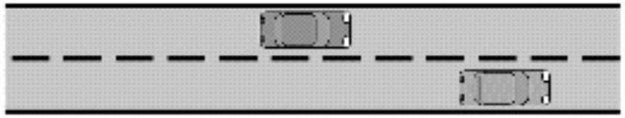
Key Concepts: The Process View

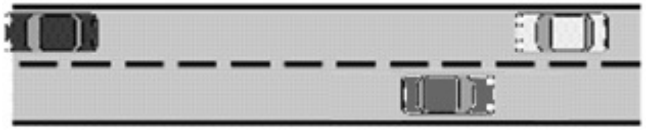


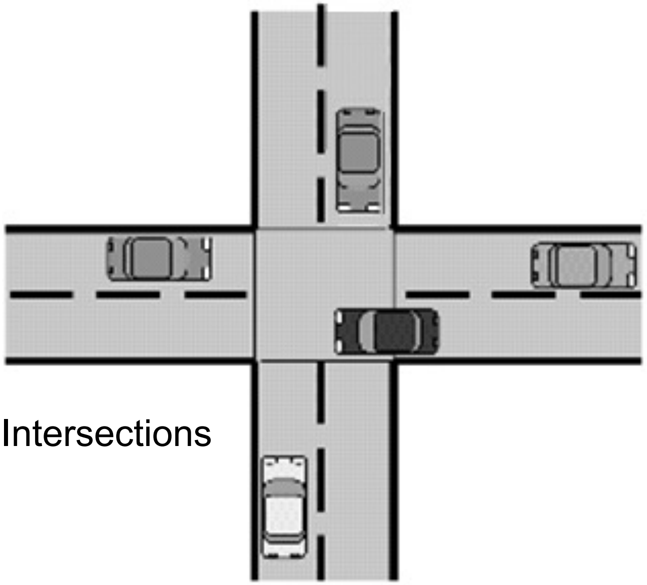
The Process View is an “architecturally significant” slice of the processes and threads of the Design Model.

What Is Concurrency?

- Example of concurrency at work:
  - Parallel roads require little coordination
  - Two-way roads require some coordination for safe interaction
  - Intersections require careful coordination

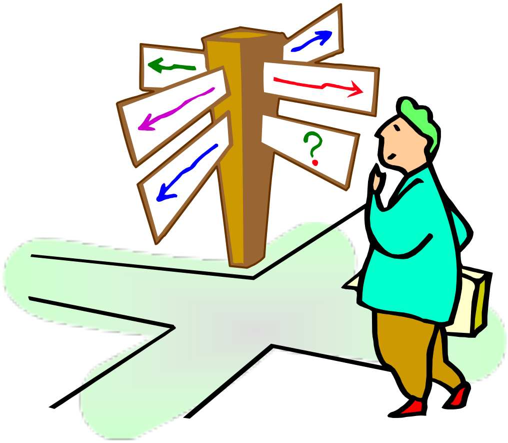

Parallel


Two-way


Intersections

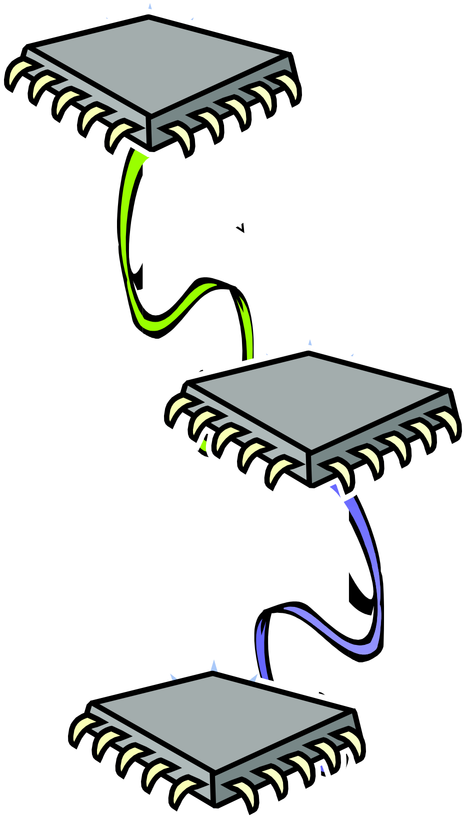
Why Are We Interested in Concurrency?

- Software might need to respond to seemingly random externally generated events
- Performing tasks in parallel can improve performance if multiple CPUs are available
  - Example: Startup of a system
- Control of the system can be enhanced through concurrency



Realizing Concurrency: Concurrency Mechanisms

- To support concurrency, a system must provide for multiple threads of control
- Common concurrency mechanisms
  - Multiprocessing
    - Multiple CPUs execute concurrently
  - Multitasking
    - The operating systems simulate concurrency on a single CPU by interleaving the execution of different tasks
  - Application-based solutions
    - the application software takes responsibility for switching between different branches of code at appropriate times



Describe the Run-time Architecture Steps

- Analyze concurrency requirements
- Identify processes and threads
- Identify process lifecycles
- Map processes onto the implementation
- Distribute model elements among processes

Concurrency Requirements

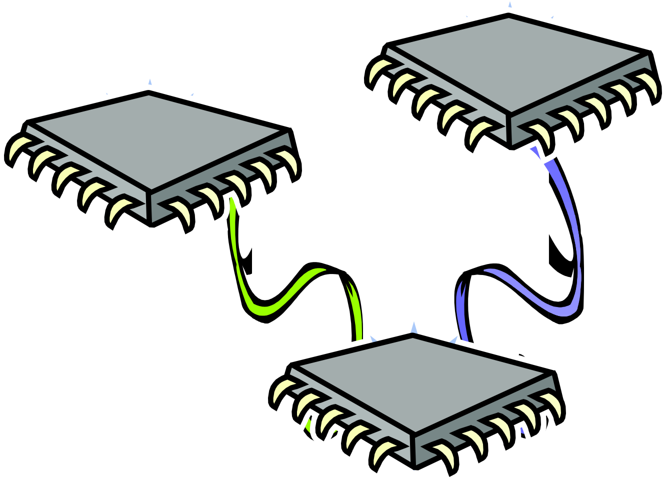
- Concurrency requirements are driven by:
  - The degree to which the system must be distributed.
  - The degree to which the system is event-driven.
  - The computation intensity of key algorithms.
  - The degree of parallel execution supported by the environment
- Concurrency requirements are ranked in terms of importance to resolve conflicts.

Example: Concurrency Requirements

- In the Course Registration System, the concurrency requirements come from the requirements and the architecture:
  - Multiple users must be able to perform their work concurrently
  - If a course offering becomes full while a student is building a schedule including that offering, the student must be notified
  - Risk-based prototypes have found that the legacy course catalog database cannot meet our performance needs without some creative use of mid-tier processing power

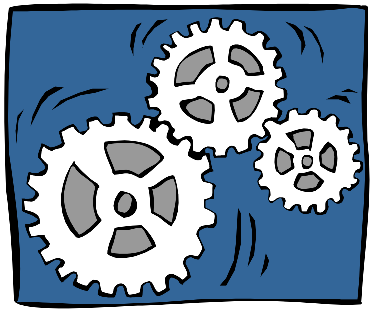
Describe the Run-time Architecture Steps

- Analyze concurrency requirements
- ☆ Identify processes and threads
- Identify process lifecycles
- Map processes onto the implementation
- Distribute model elements among processes

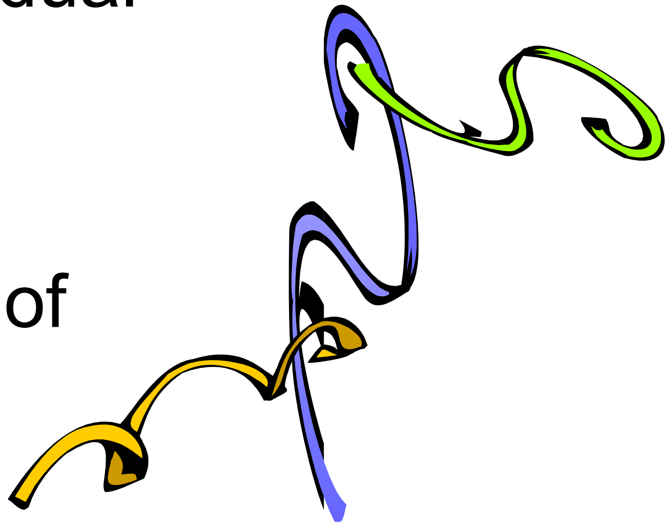


Key Concepts: Process and Thread

- Process
  - Provides heavyweight flow of control
  - Is stand-alone
  - Can be divided into individual threads

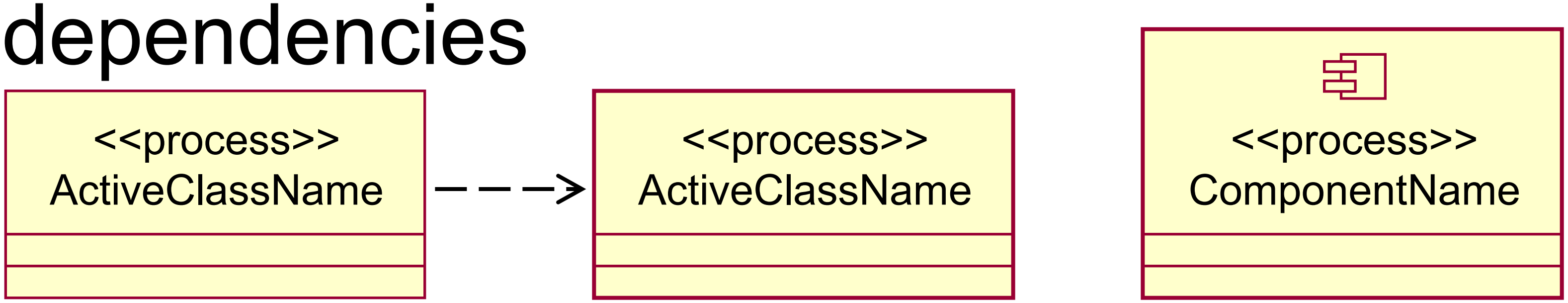


- Thread
  - Provides lightweight flow of control
  - Runs in the context of an enclosing process



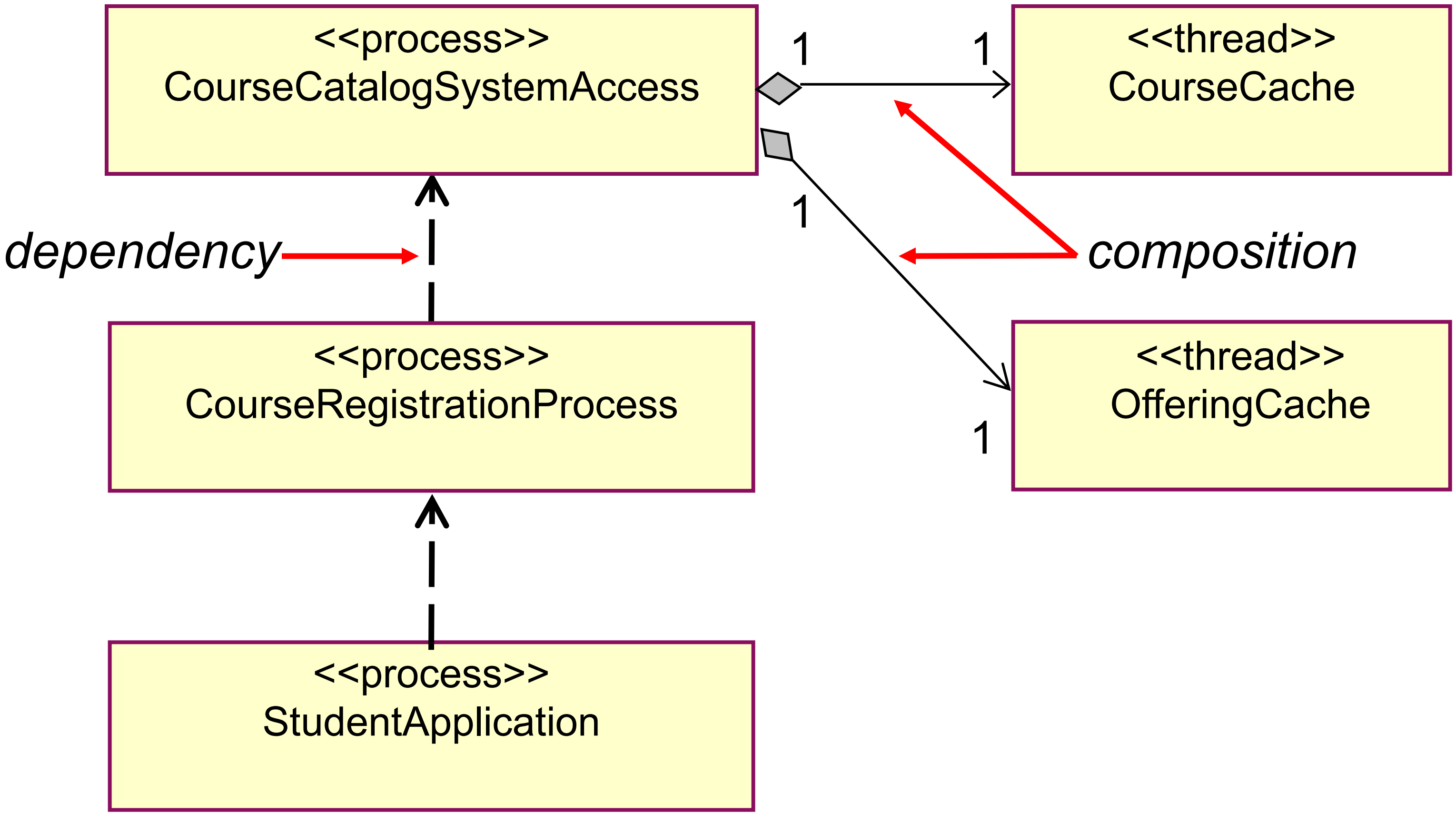
Modeling Processes

- Processes can be modeled using
  - Active classes (Class Diagrams) and Objects (Interaction Diagrams)
  - Components (Component Diagrams)
- Stereotypes: <<process>> or <<thread>>
- Process relationships can be modeled as dependencies



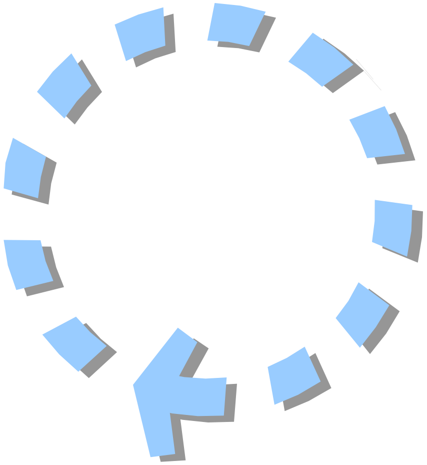
This course will model processes and threads using Class Diagrams.

Example: Modeling Processes: Class Diagram



Describe the Run-time Architecture Steps

- Analyze concurrency requirements
- Identify processes and threads
- ☆ Identify process lifecycles
- Map processes onto the implementation
- Distribute model elements among processes

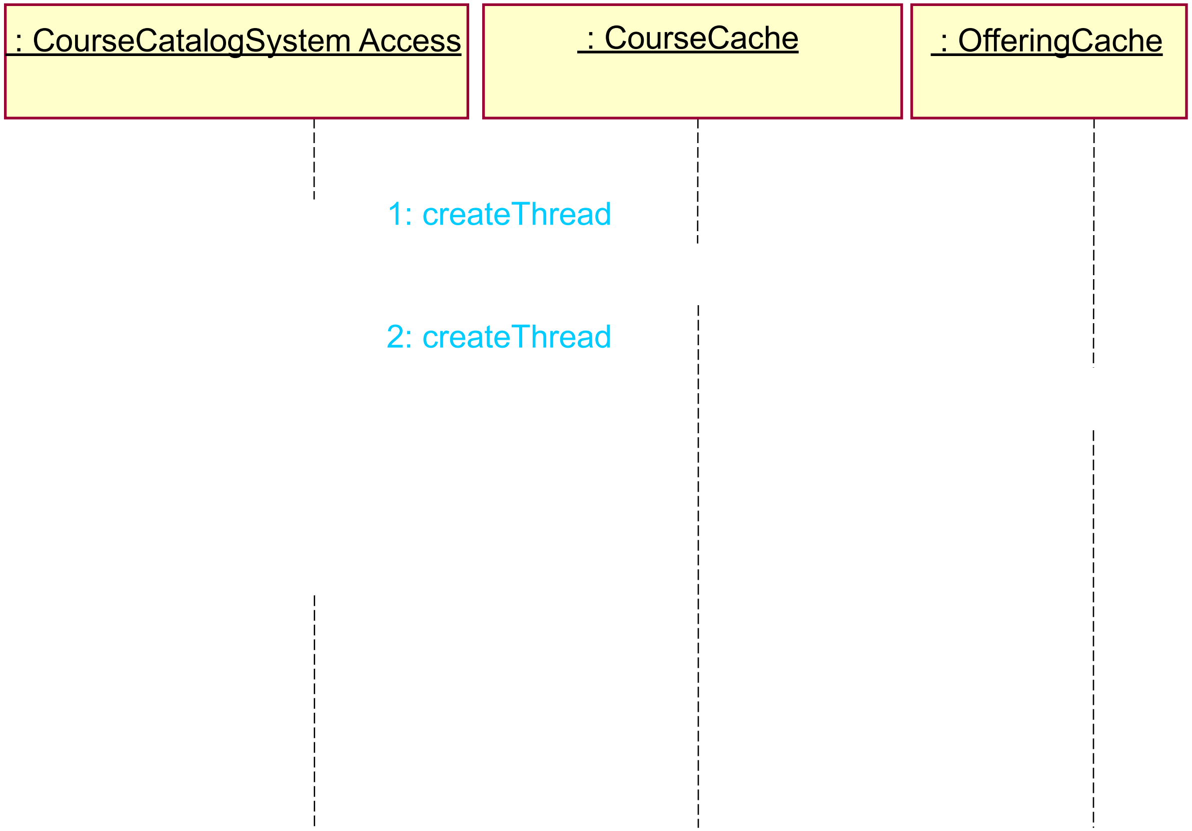


Creating and Destroying Processes and Threads

- Single-process architecture
  - – Process creation takes place when the application starts
  - – Process destruction takes place when the application ends
- Multi-process architecture
  - – New processes are typically created from the initial process that was created when the application was started
  - – Each process must be individually destroyed

Note: The Course Registration System utilizes a multi-process architecture.

Example: Create Processes and Threads



Creation of threads during application startup.

Describe the Run-time Architecture Steps

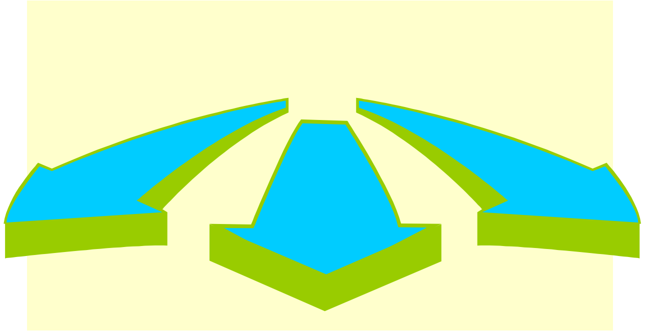
- Analyze concurrency requirements
- Identify processes and threads
- Identify process lifecycles
- ☆ Map processes onto the implementation
- Distribute model elements among processes

Mapping Processes onto the Implementation

- Processes and threads must be mapped onto specific implementation constructs
- Considerations
  - Process coupling
  - Performance requirements
  - System process and thread limits
  - Existing threads and processes
  - IPC resource availability

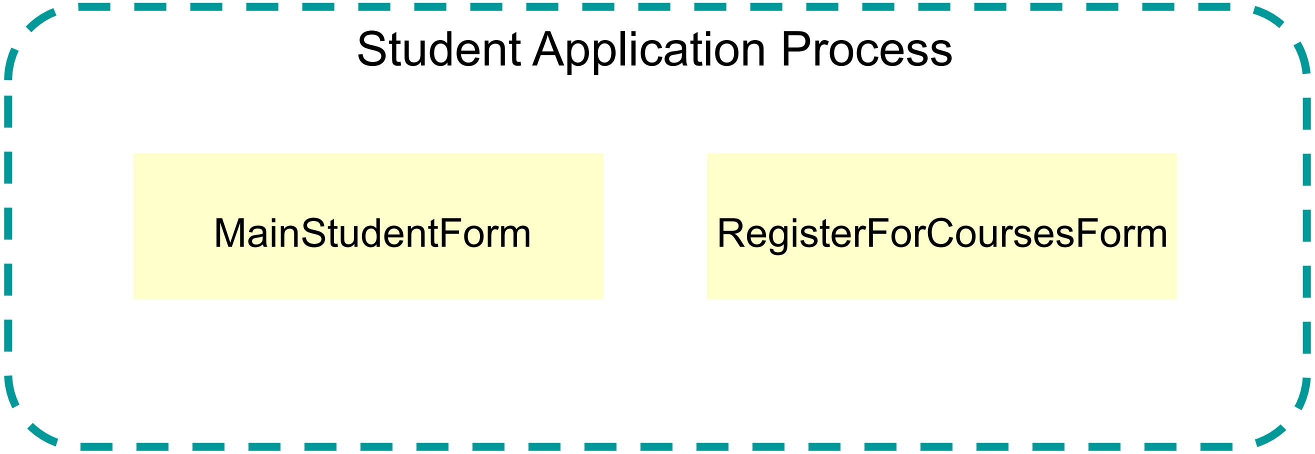
Describe the Run-time Architecture Steps

- Analyze concurrency requirements
- Identify processes and threads
- Identify process lifecycles
- Map processes onto the implementation
- ☆ Distribute model elements among processes



Design Element Allocation

- Instances of a given class or subsystem *must* execute within at least one process
  - They may execute in several processes



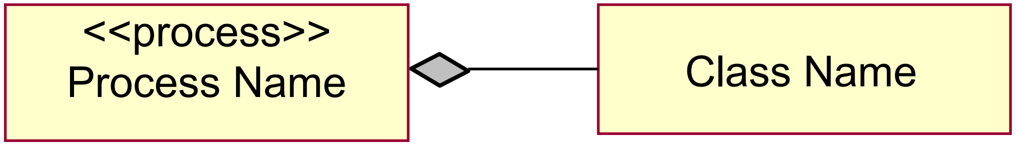
Modeling the Mapping of Elements to Processes

- Class diagrams
  - Active classes as processes/threads

<<process>>
Process Name

<<thread>>
Thread Name

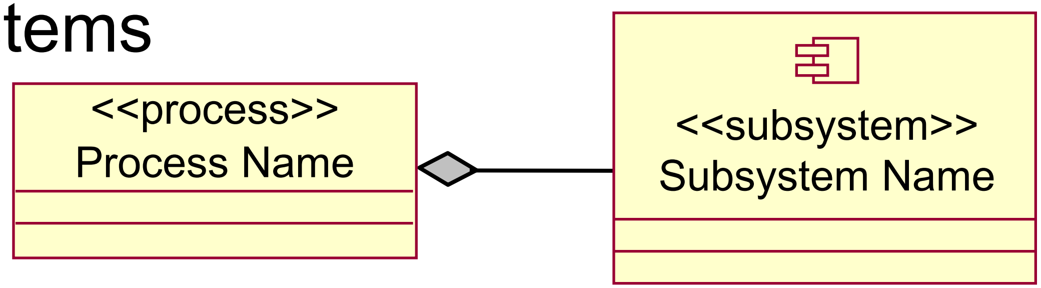
  - Composition relationships from processes/threads to classes



<<process>>
Process Name

Class Name

  - Composition relationships from processes/threads to subsystems

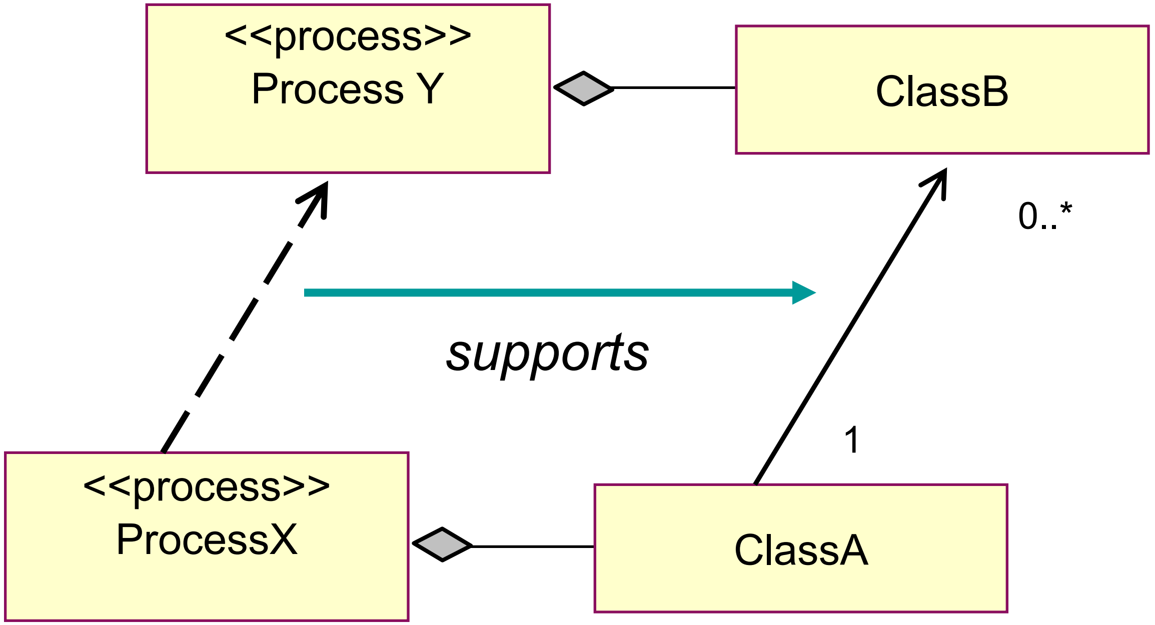


<<process>>
Process Name

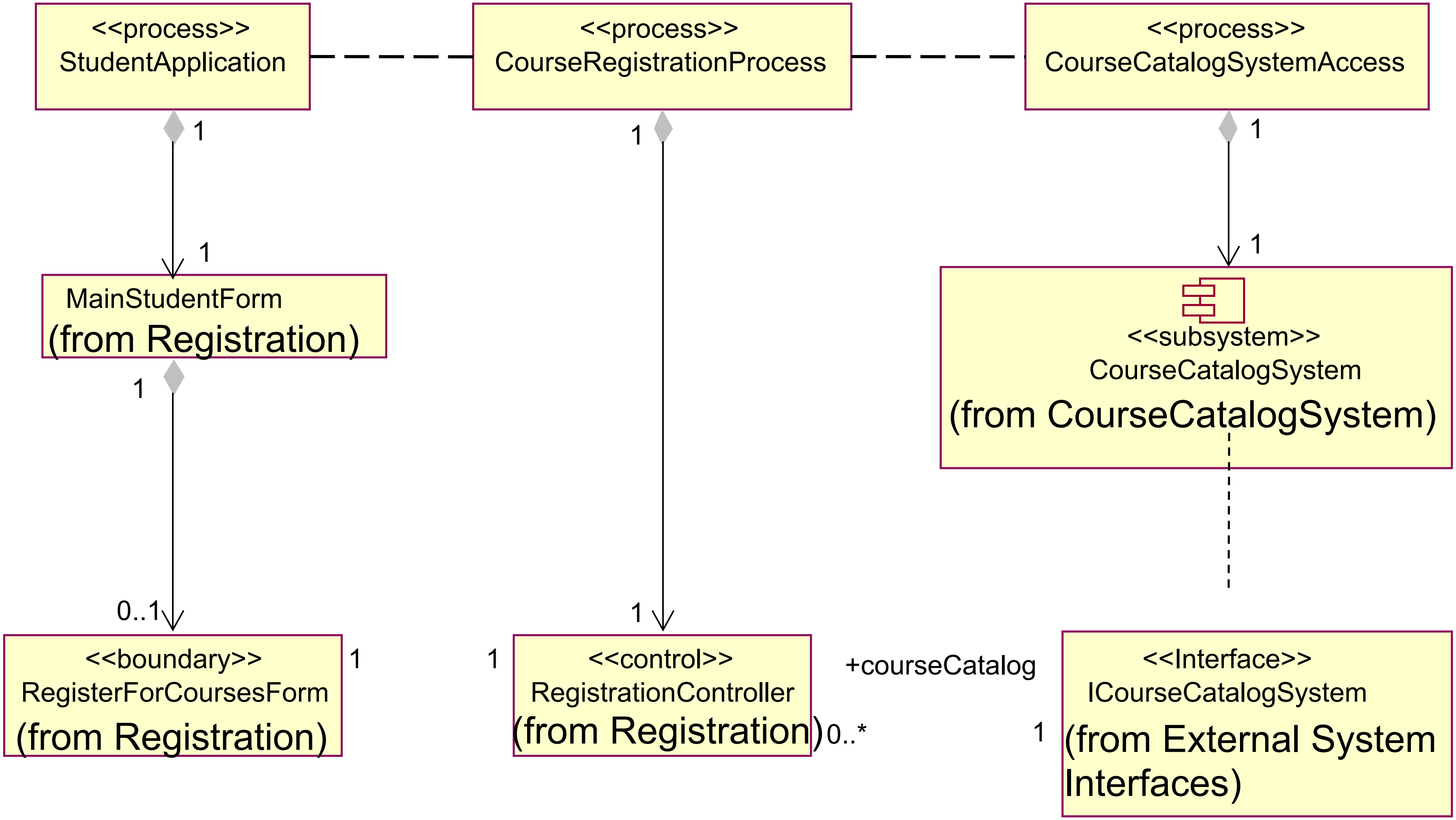
<<subsystem>>
Subsystem Name

Process Relationships

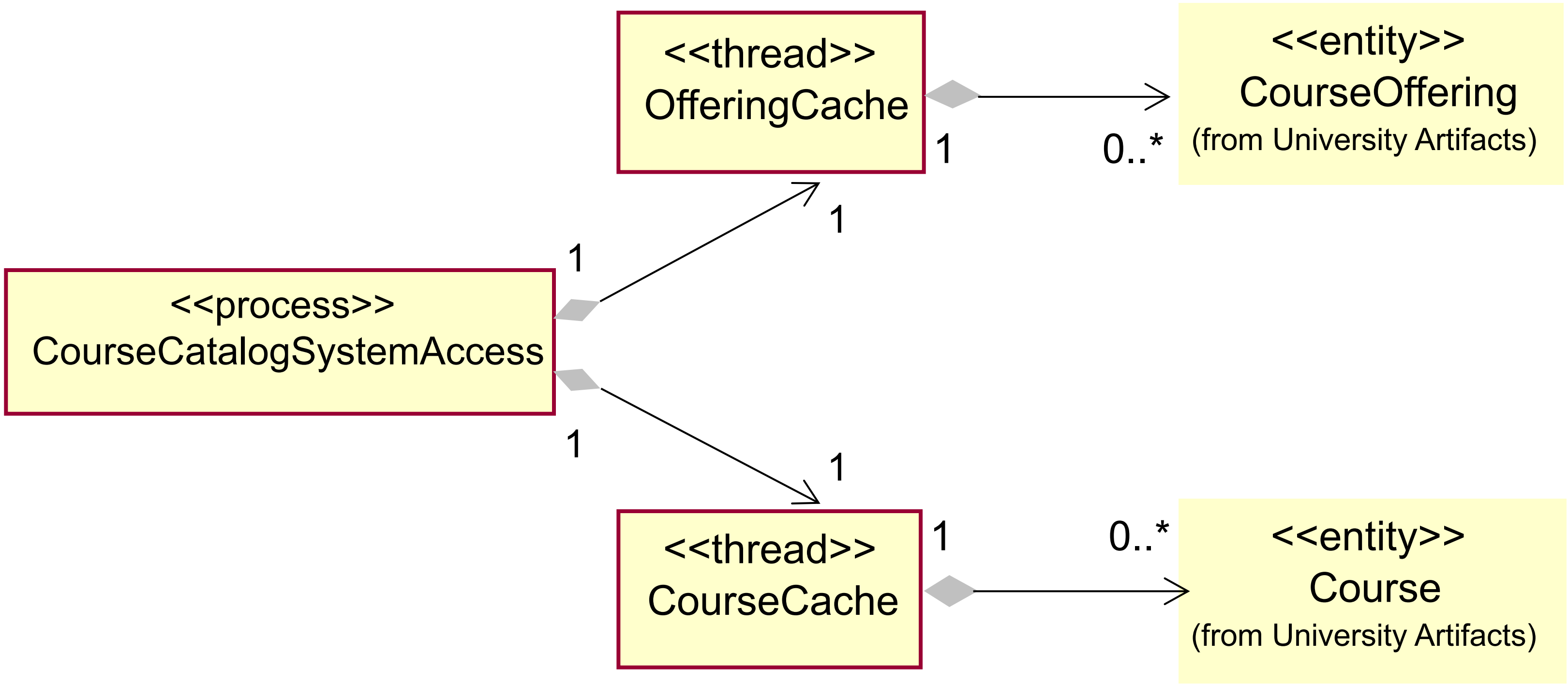
- Process relationships must support design element relationships



Example: Register for Course Processes



Example: Register for Course Processes (continued)



Checkpoints: Describe the Run-time Architecture

- Have all the concurrency requirements been analyzed?
- Have the processes and threads been identified?
- Have the process life cycles been identified?
- Have the processes been mapped onto the implementation?
- Have the model elements been distributed among the processes?

Review: Describe the Run-time Architecture

- What is the purpose of the Describe the Run-time Architecture activity?
- What is a process? What is a thread?
- Describe some of the considerations when identifying processes.
- How do you model the Process View? What modeling elements and diagrams are used?