

Systems Development

Requirements Analysis

<http://lbgeeks.com/gitc/systemsAnal.php>

April 2, 2007

- **Requirements And Sub Phases**
- **Why Analysis?**
- **Conflict Resolution**
- **Subsystem Decomposition**
- **IPO Diagrams**
- **Use Cases**
- **Schedule Reminders**

Requirements And Sub Phases

- **For this course, we split requirements into:**
 - **Needs assessment**
 - **Requirements specification**
 - **Requirements analysis**
- **Customer interaction goes from high to low**
- **Each sub phase addresses different results:**
 - **Determining what system should do**
 - **Defining features objectively**
 - **Checking for completeness, correctness**
- **Without all three results, we cannot design!**

Why Analysis? - 1 of 2

- **So far we have:**
 - **Captured needs, murmurs, wants from customers verbatim**
 - **Translated customer statements into engineering requirements**
- **Minor filtering so far:**
 - **Involve customer as much as possible**
 - **Capture input for discard later**

Why Analysis? - 2 of 2

- **Not all needs, murmurs, wants are possible:**
 - **Technical limitations**
 - **Some statements at cross purposes**
 - **Limits on budget, skills, schedule**
- **Goals for analysis:**
 - **Eliminate conflicts among requirements**
 - **Determine top level system partitioning**
 - **Model interfaces, or inputs, activities, outputs among subsystems**
 - **Communicate delivered functions to customers**

Conflict Resolution

- **Definition**
- **Finding Conflicts**
- **Resolution Methods:**
 - **Ethical**
 - **Goal Oriented**
 - **Mathematical**

Conflict Definition

- **Mutual exclusions among two or more different requirements**
- **Various levels of severity:**
 - **Personal style or taste**
 - **Duplication of effort or resources**
 - **Technical impossibility**
 - **Danger to life or property**
- **Resolve prior to start of design**
- **One conflict may break a single design into different directions**

Finding Conflicts

- **Consider all requirements as a group**
- **Find failure conditions:**
 - **Output to input alignment failure**
 - **Duplicate or oppositional processing**
 - **Resource contention**
 - **Order or timing dependency mismatches**
- **Note identifiers of colliding requirements**
- **Resolution is simply choosing a single path out of the collision**

Ethical Resolution

- **Choice of path based on ethics or morals:**
 - Does choice provide maximum benefit?
 - What about minimum harm?
- **Following aspects considered:**
 - Trust between parties
 - Organization position and reputation
 - Group before self
- **Classifications:**
 - Right vs. right
 - Right vs. unknown wrong
 - Right vs. known wrong

Goal Oriented Resolution

- **Maximize objectives achieved by the system**
- **Levels:**
 - **High-level, strategic concerns**
 - **Low-level, technical concerns**
- **Types:**
 - **Function**
 - **Quality**
 - **Objective**
 - **Subjective**
- **Goals, like requirements, have a taxonomy**
- **Need same precision in specification**

Mathematical Resolution

- **Conflicts occur at set union**
- **Four types:**
 - **RR, or specification failure in both sets**
 - **II, or interpretation failure in both sets**
 - **IR, or interpretation and specification**
 - **RI, or specification and interpretation**
- **RR conflicts require mutual agreement**
- **II can be resolved with better requirements**
- **IR or RI means one set or the other must adapt**

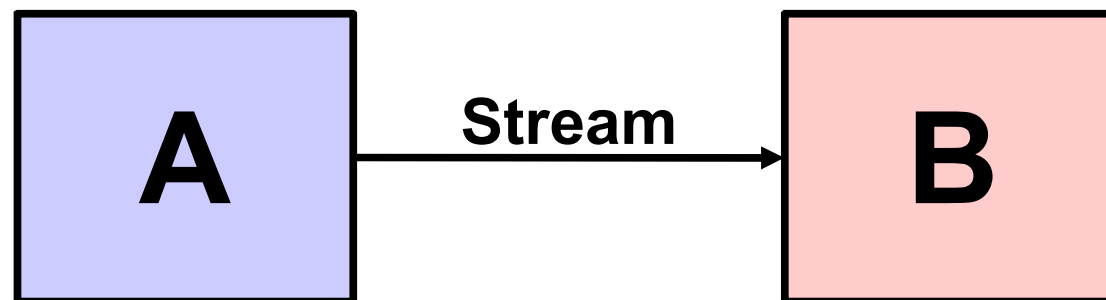
Subsystem Decomposition

- **Definitions**
- **Techniques:**
 - **Streams**
 - **Formats**
 - **Complexity**
- **Input-Process-Output Diagram**
- **Examples**

- **Subsystem:**
 - Collection of related requirements
 - Component within the system
 - Defined interface (input-output boundary)
 - Performs I/O transformation (activity, process)
- **Decomposition:**
 - Opposite of compose
 - Break down into smaller pieces
 - Transform from monolith into separate entities
- **Need techniques to suggest boundaries**

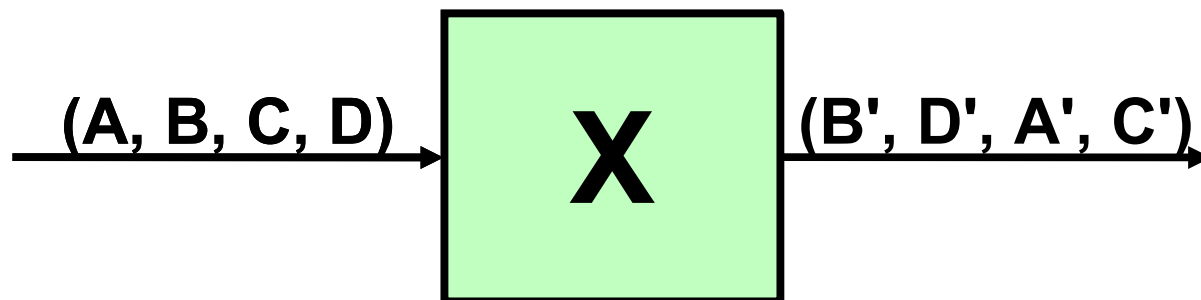
Stream Technique

- **Does 100% of output from A feed B?**
- **If so, we have producer-consumer relation**
- **Elements generating A and absorbing B are subsystems**



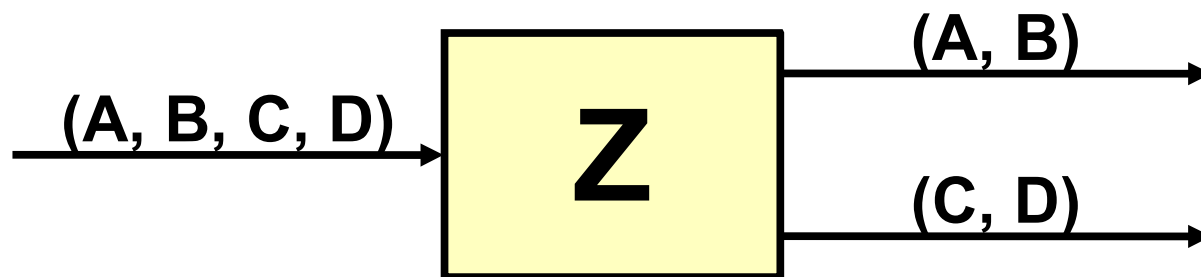
Format Technique

- **Is the input of a node identical to its output except for a defined transformation?**
- **If so, we have formatting process**
- **Element performing transformation might be a subsystem**



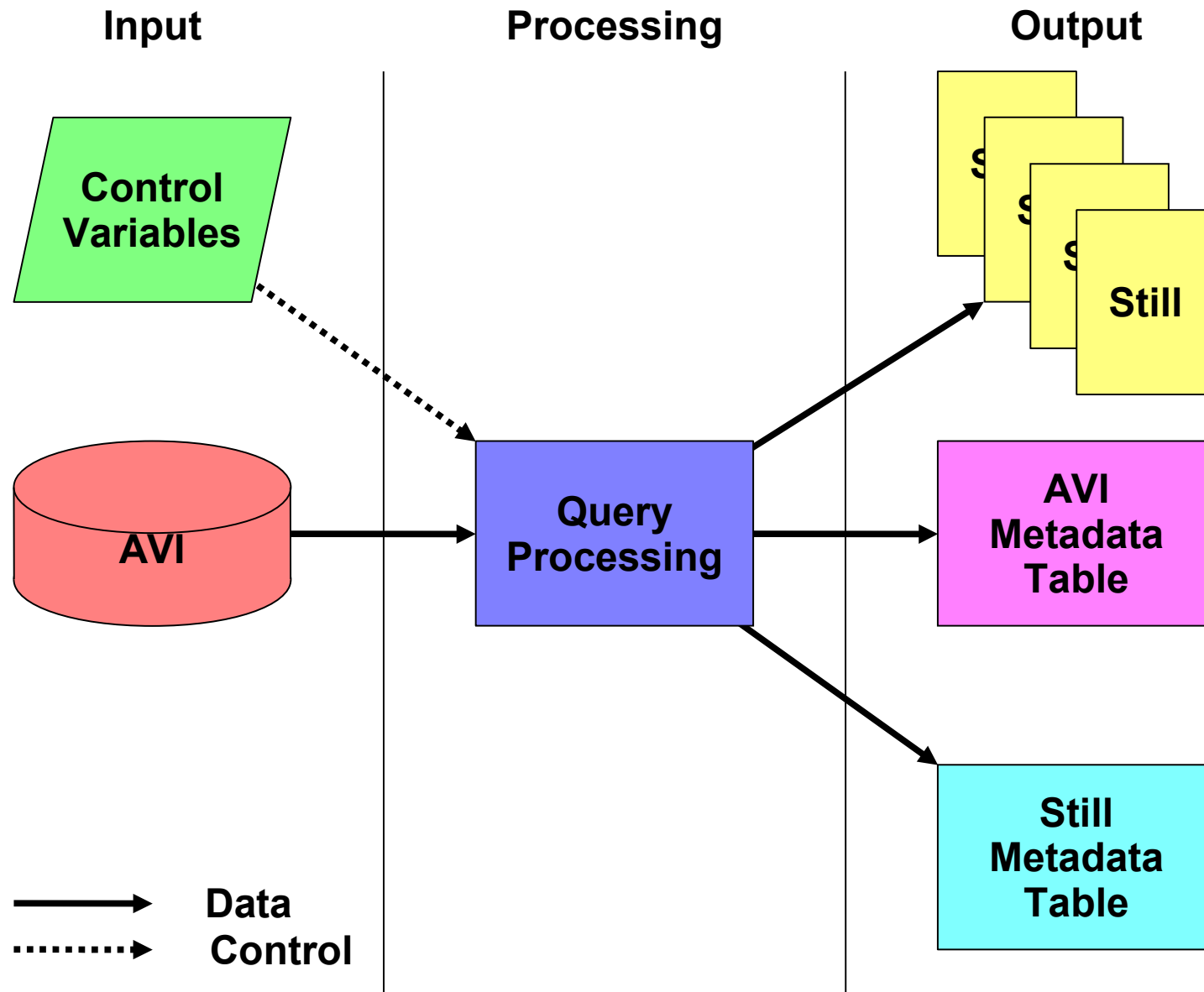
Complexity Technique

- **Do we split or combine a set of data?**
- **If so, we introduce a complex process**
- **Elements splitting and joining set may be subsystems**



- Comes from Input-Process-Output
- Documents interface:
 - Across one subsystem
 - Among multiple subsystems
- Inputs can be control or data
- Text necessary to explain the I, P, and O
- More requirement focused than object or sequence diagrams

IPO Diagram Example

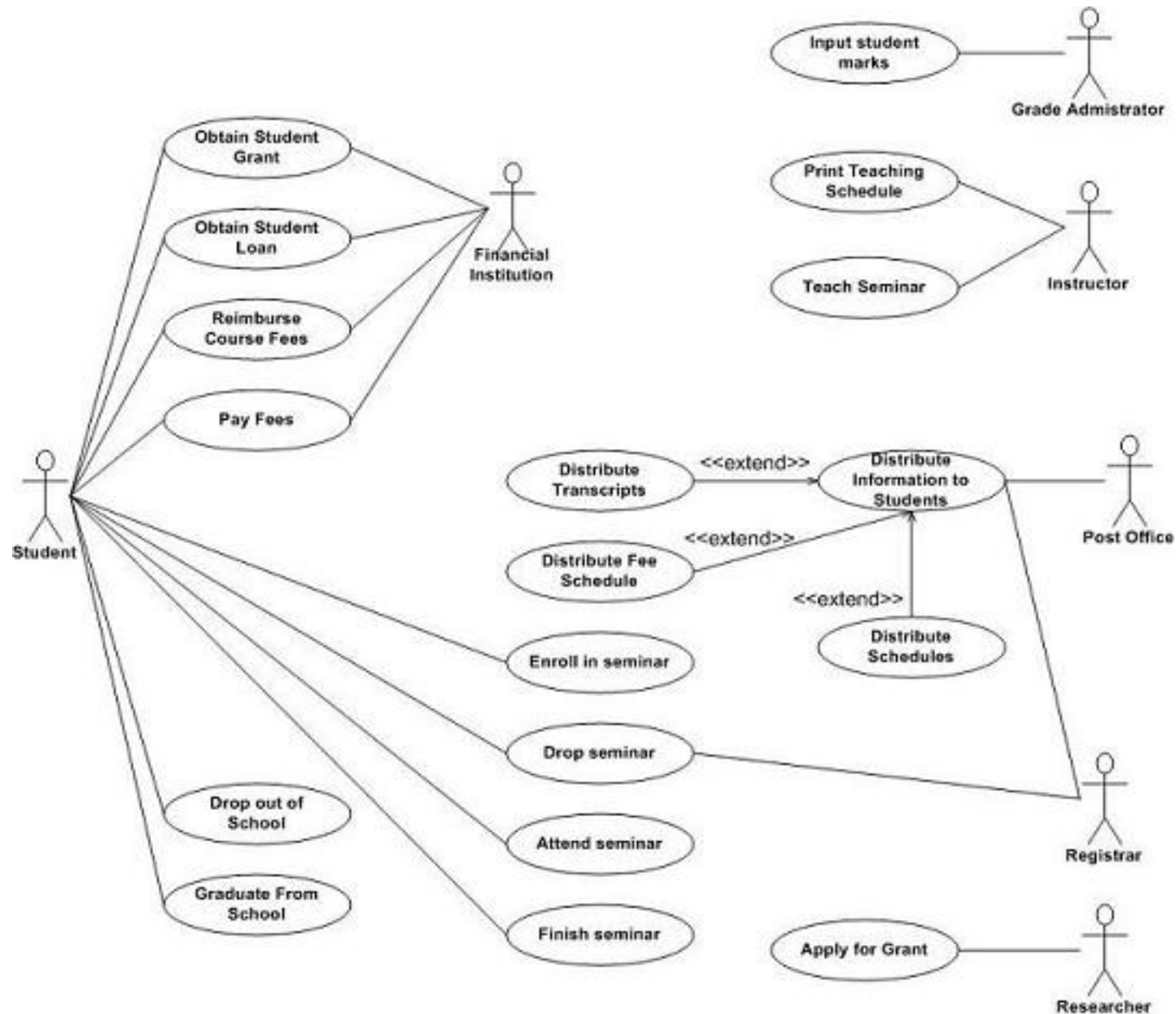


- **Definitions**
- **Notations**
- **Example**

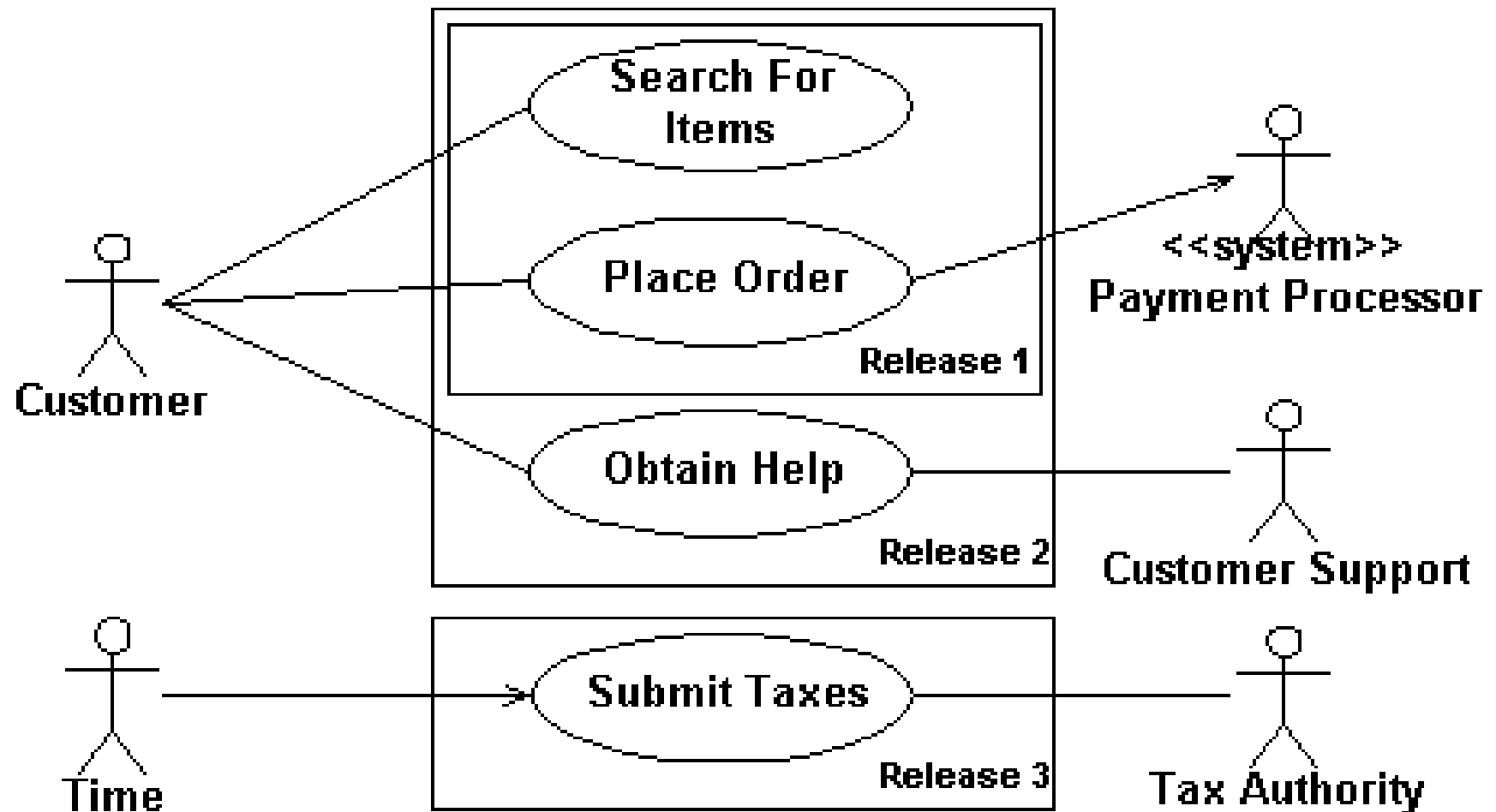
- **Actor**
 - **Person using system**
 - **Another external system**
- **Use case**
 - **Transaction run with system**
 - **Initiated by input from actor**
 - **Concludes with output to actor**

- **Actors are stick figures on diagram**
- **Use cases are ovals**
- **Solid arrows or lines mean association**
- **Boxes indicate subsystem boundaries**
- **Packages represent file folders**
- **Ordering from top to bottom significant**

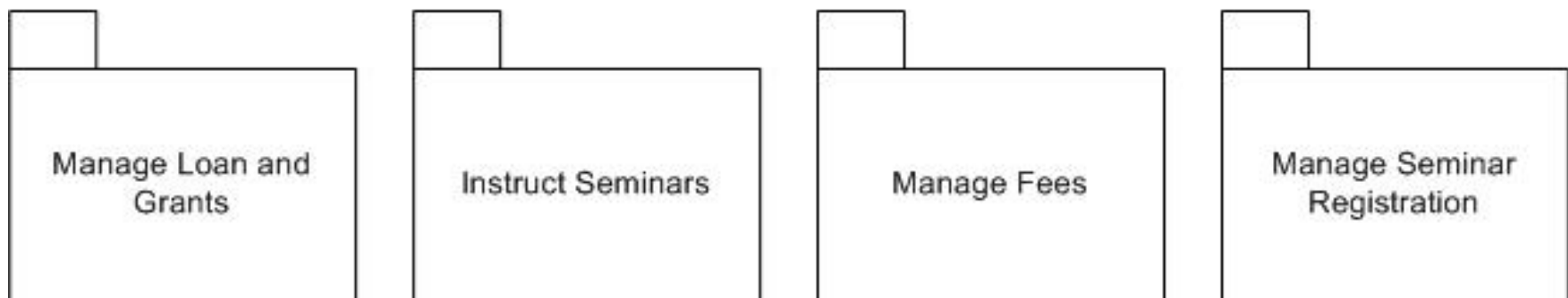
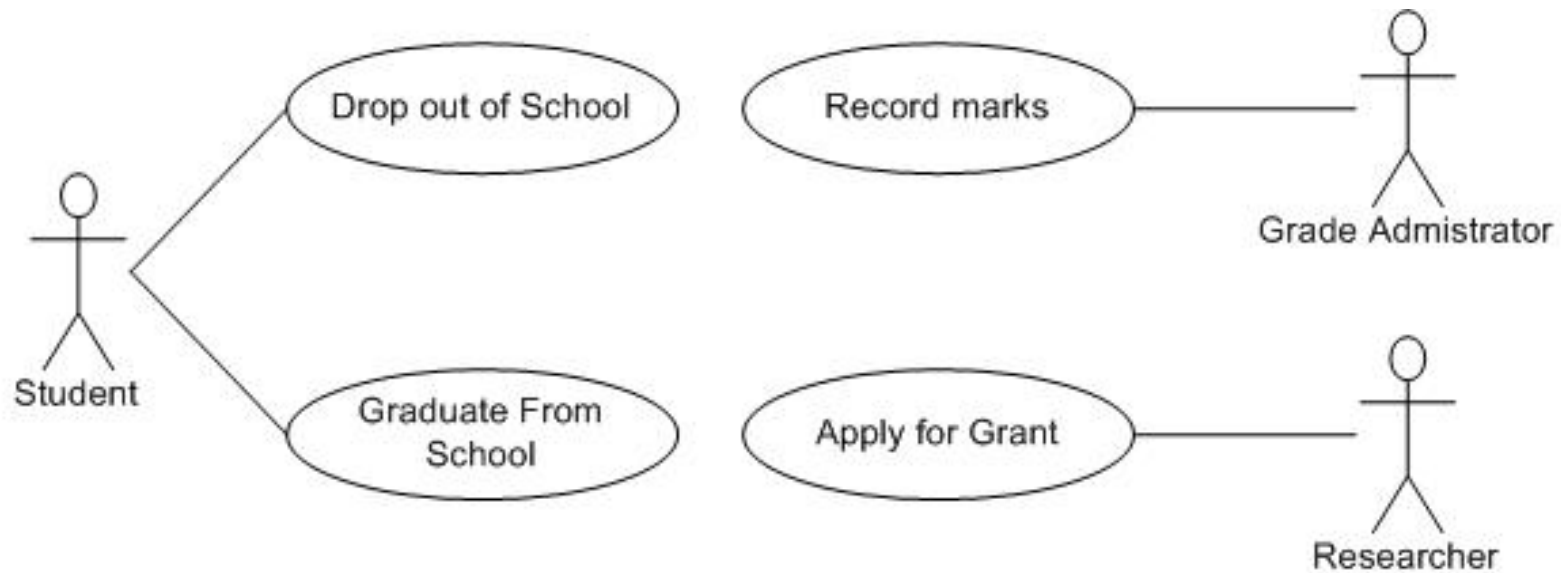
Example – Use Case



Example – Boundaries



Example – Packages



Class Schedule Reminders

- **Sessions at 19:00:**
 - **Wednesday, April 4**
 - **Monday, April 9**
 - **Wednesday, April 11**
- **Next Unit Starts April 16**
 - **Quiz at 16:45**
 - **Assignment due at start of class**