OOA
• Define classes, their relationships and their behaviors
• Define class
  – Attributes
  – Methods (behaviors/operations/responsibilities)
• Define class hierarchy

Scenario Based Modeling
• Develop use-case(s)/scenarios and use-case diagram
• UML activity diagram
  – Supplements use-cases
  – Provides graphical representation of the flow of the interaction
  – Symbols
    • Rounded rectangles – system function
    • Directed arcs – flow through the system
    • Diamond – branching decision
    • Horizontal lines – parallel activities
  – Adds more information by including constraints explicitly
• UML Swimlane Diagram
  – Adds actor responsibility to activity diagram when multiple actors are involved in a use case
  – Partition the diagram into areas for each actor involved and place their functionality in their partition
enter password and user id

Valid user id/password

terminate session
choose drop classes
choose add classes

return to previous menu

Invalid user Id/password
Prompt for reentry

Tries remain

enter course number

view course data
display course does not exist
drop course

UML Activity Diagram
Register for a Class
UML Swimlane Diagram
Register for a Class
OOA -- Class Based Modeling
• Determine classes, attributes and methods (operations/responsibilities)

Identifying Classes
• Grammatical parse on use-case text or processing narratives
• Nouns or noun clauses
• Look for
  - External entities
    . Produce or consume information
  - Things
    . Reports or displays, etc.
  - Occurrences or events
  - Roles
  - Places – establish context
  - Structures
• Coad/Yourdon selection criteria
  - Satisfy almost all of these
  - Retained info
  - Needed service
  - Multiple attributes
  - Common attributes
  - Common operations
  - Essential requirements

Specify Attributes

Define Operations
• Manipulate
• Perform calculation
• Inquire about state
• Monitor for a controlling event
• Verbs in the grammatical parse

CRC Cards
• Class-responsibility-collaborator
• Index cards
• top class name
• Left responsibilities
• Right collaborators
  – Can the class fulfill the operation on its own or does it need help from another class

Review CRC model
• Representatives of customer and developer
• Each participant given a subset of cards such that a single representative does not have any collaborators
• Read use case
  – When come to a class see if it can fulfill the task
  – Call on collaborators as needed
• Modify the model as needed to ensure all use cases can be carried out

Associations and Dependencies
• Relationships between classes
  – Shows multiplicity (cardinality)

Analysis Packages
• Categorize the classes into unit of corresponding nature