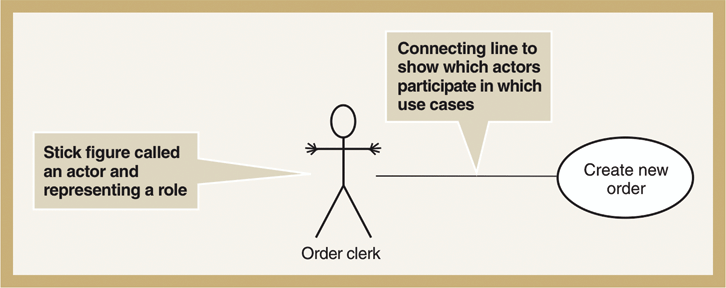
Functional Modeling

**Requirements Challenge**

* Toughest part in a system design is to elicit the correct and necessary system requirements from the stakeholders

**Use case modeling**

* Process of modeling a system’s functions in terms of **business events, who initiated that event, and how the system responds to the events**

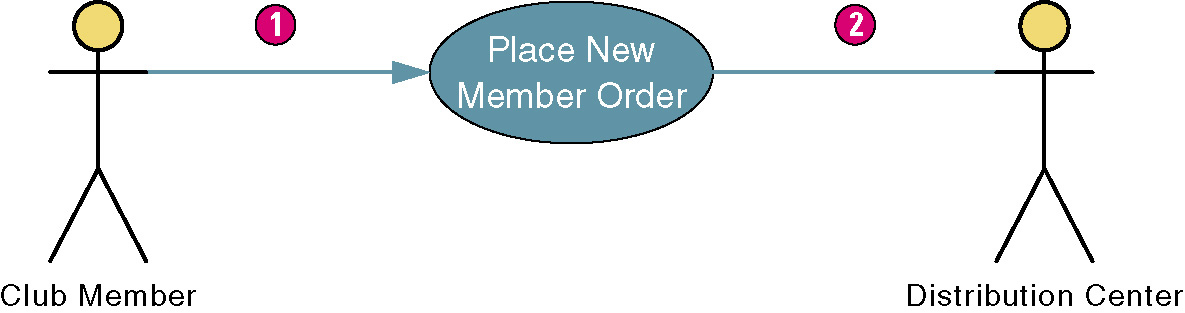
****

**Use-Case modeling**

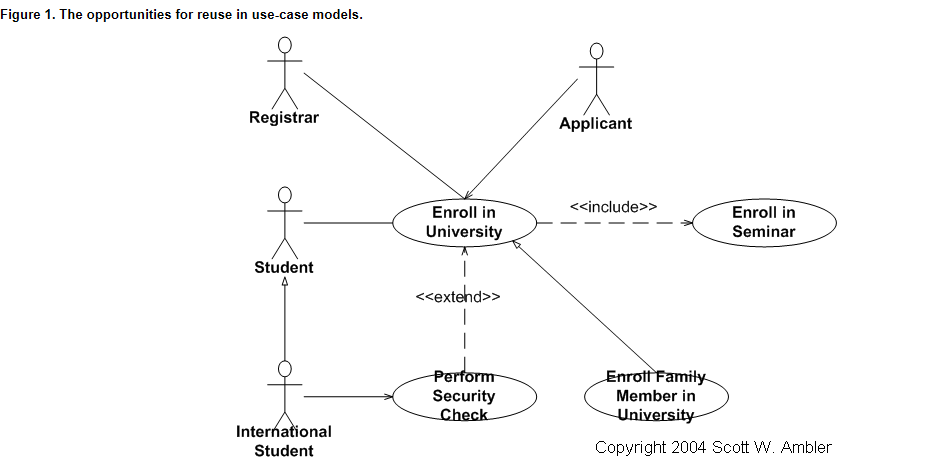
* Use-Case Diagram
  + A diagram that depicts interactions between the system, external systems and users.
  + Graphically represents who uses the systems and what ways the users interact with them.
* Use-Case Description
  + Textual description of the business event and how the user will interact with the system to accomplish a task
* Use-Case
  + A sequence of steps for completing a single business task
  + Represents a major piece of system functionality
  + Subset of overall system functionality
  + Describes the system’s functions from the perspective of an external user.
* Actor
  + A person or system that derives benefit from and is external to the subject.
  + Labeled with role (noun)
  + Anything that interacts with the system to exchange information
  + Can be associated with other actors
  + Types
    - Primary business actors
      * The stakeholder the primarily benefits from the execution of the use case (employee receiving a paycheck)
    - Primary system actor
      * Stakeholder that “triggers” the event (bank teller deposit check)
    - External server actor
      * Stakeholder that responds to a request from a use case (credit bureau authorizing a credit card exchange)
    - External receiver actor
      * Stakeholder that is not primary actor but receives something of value (warehouse receiving a packing slip)
* Illustrates the activities that are performed by users of a system
* Describes basic functions of a system
  + What a user can do
  + How the system responds
* **Benefits:** 
  + Serves as a tool for capturing functional requirements
  + Helps to breakdown system into more manageable pieces
  + Provides a means of communicating with users and other stakeholders to make sure everything is properly understood
  + Serves as aid to determine project scope, effort and schedule
  + Provides a baseline for testing in terms of defining test plans/cases

**Use Case Syntax**

* Relationships
  + Association (----------)
    - A relationship between an actor and a use case where an interaction happens between them.
    - Can be bidirectional or unidirectional
    - Solid line connecting actor to use case
    - (1) is the initiator
    - (2) is the receiver



* + Include (- - - - - - - -> <<includes>>)
    - Includes mean that a functionality can be re-used
    - It can be use for multiple cases
    - Indicates a dependency on another use case
  + Extend (- - - - - - - -> <<extends>>)
    - Adds functionality it does not re-use any functionality.
    - Indicates
      * Adding new features/capabilities to a use case
      * Any optional use cases
    - Used to add steps to a primary use case
  + Generalization ( ----------------->) aka inheritance
    - Indicates common behaviour between two actors.
    - Student and international student below
    - Inherits from the interactions of its parent.

****

**Steps in Writing Use Cases**

1. **Identify the business actors**
   1. Ask the following questions to identify them
      1. Who or what provides inputs to the system?
      2. Who or what receives outputs from the system?
      3. Are interfaces required to other systems?
      4. Who will maintain that information in the system?
   2. Each actor must have a describing role
   3. Can identify goals of each actor in the description
   4. Needs a description of role
2. **Identify the major use cases**
   1. Identify system boundaries
   2. Identify and write major use cases
   3. Has a verb-noun name and a brief description
   4. Some questions to ask:
      1. What are the main tasks of the actor?
      2. What information does the actor need from the system?
      3. What information does the actor provide to the system?
3. **Expand the major use case**
   1. Choose one major use case to expand
   2. Come up with a use-case template containing flow of events
   3. Come up with happy path
   4. Determine alternative flows
   5. Create sub-flows
   6. Simplify
4. **Confirm the major use case**
   1. Review the current set
   2. Involve users to get feedback
   3. Iterate the entire set of steps until all use cases are defined
5. **Create the use case diagram**
   1. Start with system boundary
   2. Place use cases on the diagram
   3. Place actors on the diagram
   4. Connect the actors with association lines