More UML, Use-case diagrams, and CRC cards

CS151

Chris Pollett

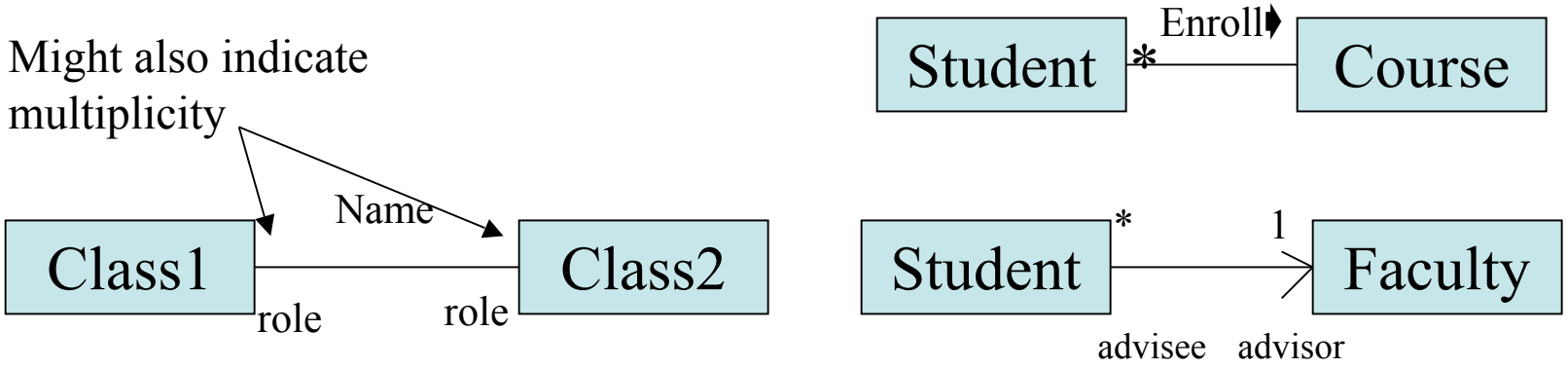
Aug. 31, 2005.

Outline

- UML for associations, aggregation, composition, and dependency
- Sequence Diagrams
- State Diagrams
- Use Cases
- CRC Cards

UML for Associations

- Associations represent general binary relationships between classes.
- One or both classes typical has a direct or indirect reference to the other class.



More On Multiplicities

- Can be one of

  l..u a number between l and u

  i a single number (can separate with a comma)

  * 0 or more

- For example, 1..* would mean 1 or more. 1,4,5 would mean 1 or 4 or 5.

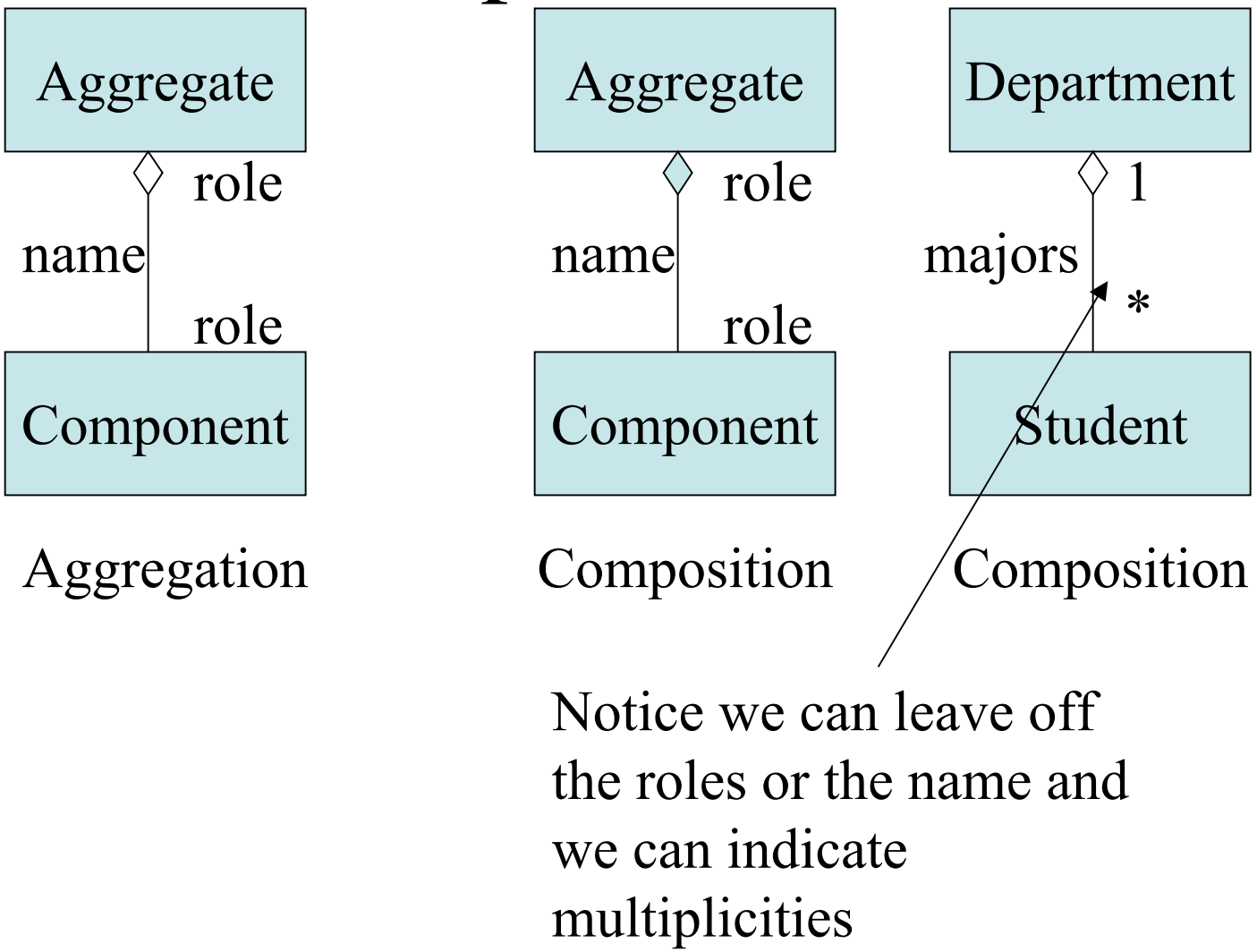
Navigation

- An association may indicate one of the two classes can invoke a member of the other class directly or indirectly.
- For example, a Creature object might have a method parent() which returns the World it is in. This in turn might have a method getGraphicsContext() which return a GraphicsContext. So Creature might be associated with GraphicsContext.

Aggregation and Composition

- These are special forms of associations.
- Aggregation is used to represent has-a or part-whole relationships Ex: Bicycle has a Tire
- Composition is a type of aggregation where the component only belongs to the particular kind of whole.
- For example, as Tire's could be Car or Bicycle Tire's they are not an example of a composition.
- It might be that a College can only exist as part of a University in which case, we'd have an example of composition.

UML for Aggregation and Composition



UML for a Dependency

- A dependency is a relationship between entities such that the proper operation of one entity depends on the presence of the other entity.

Indicates Class1 depends on Class2

Modeling Dynamic Behavior

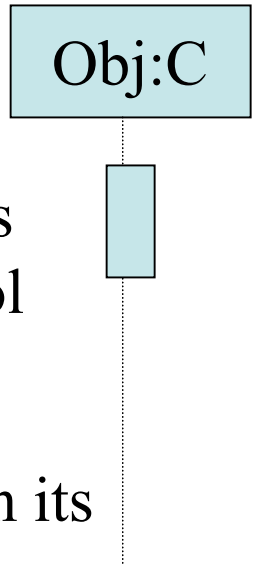
We'd now like to discuss two ways to model the behavior of classes and objects as a software program is running:

- Sequence Diagrams
- State Diagrams

Sequence Diagrams

Consist of set of vertical columns of the form:

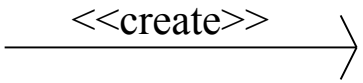
An object



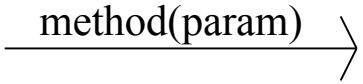
has focus of control

for some period on its lifeline

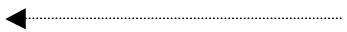
Together with different kinds of arrows between these columns:



Creating an object

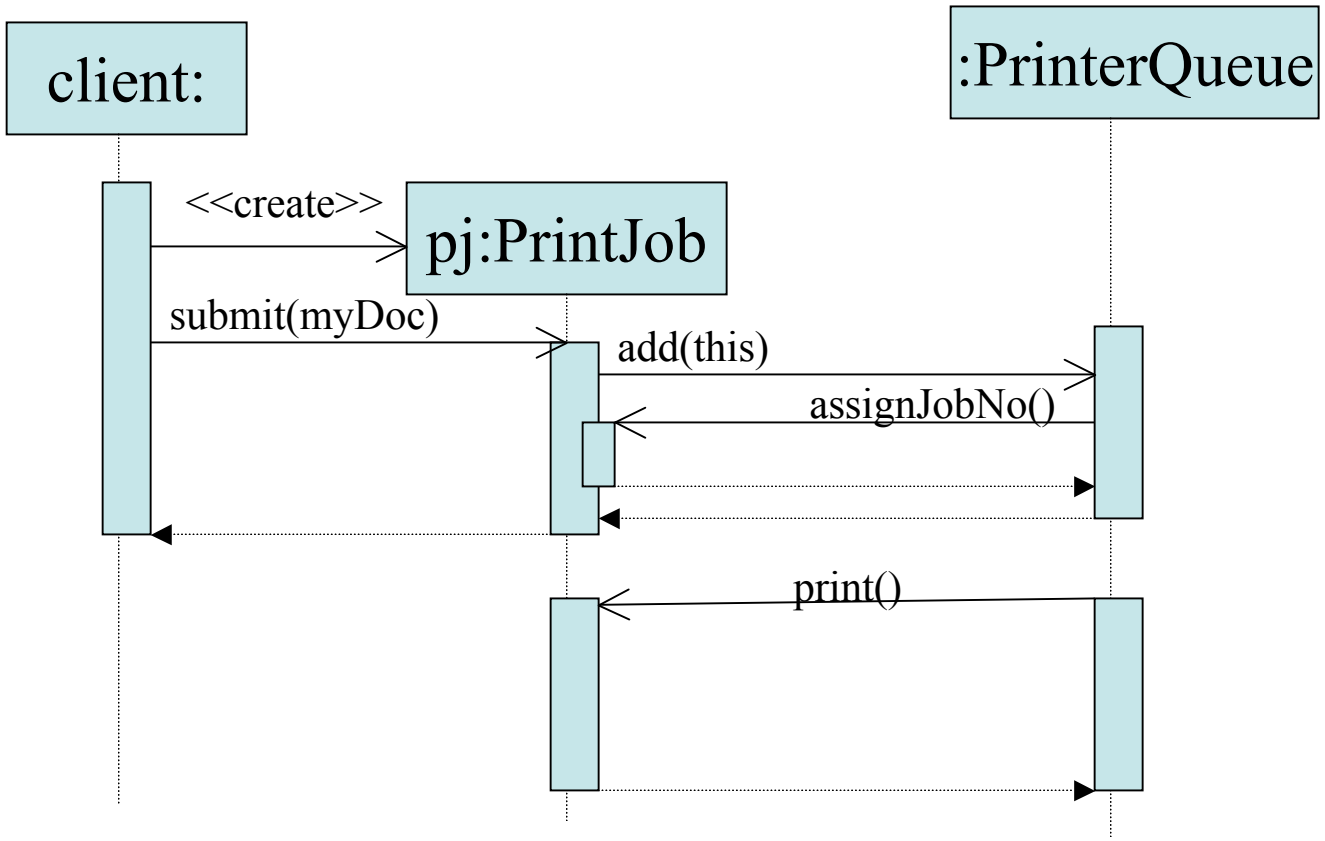


Invoking a method



Returning from a method

Example Sequence Diagram



State Diagrams

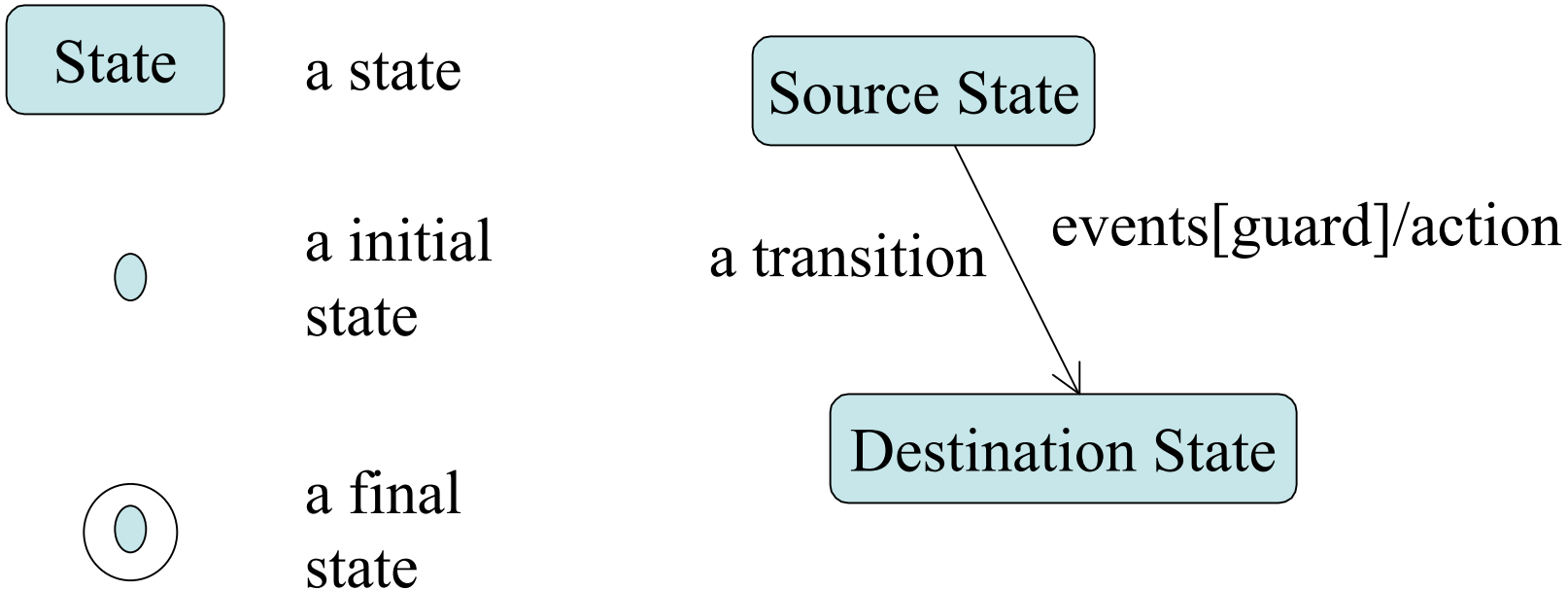
- Depict the flow of control in a software system using the concepts of states and transitions between states.
- Generalize finite state machines from CS154.

A *state* is a condition or situation an object might be in at given time.

A *transition* is a relationship between two states indicating the object started in the first state, performs some action, and ends up in the second state. A transition may have a triggering event. In addition, transitions may have an action that occurs with them.

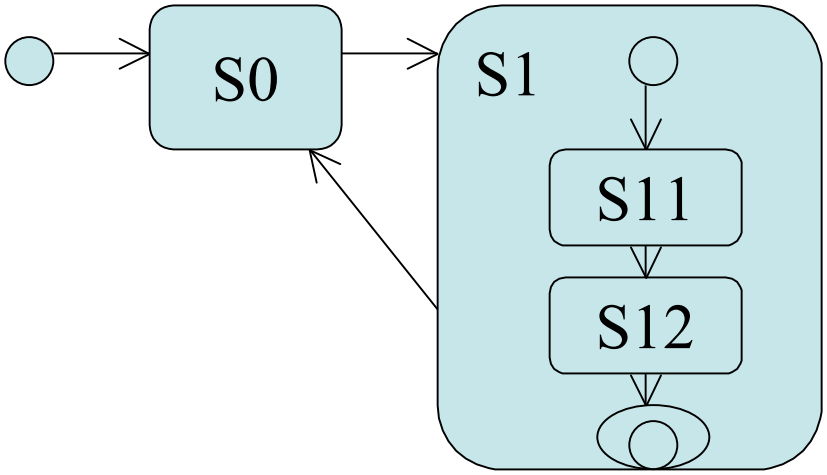
An object begins in an *initial state* goes through some sequence of intermediate states and ends in a *final state*.

UML for State Diagrams

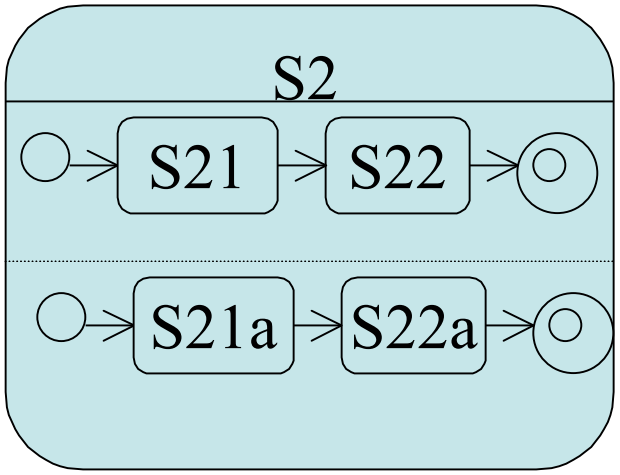


Nested State Diagrams

- States can be nested to make composite states:



(a) composite sequential states

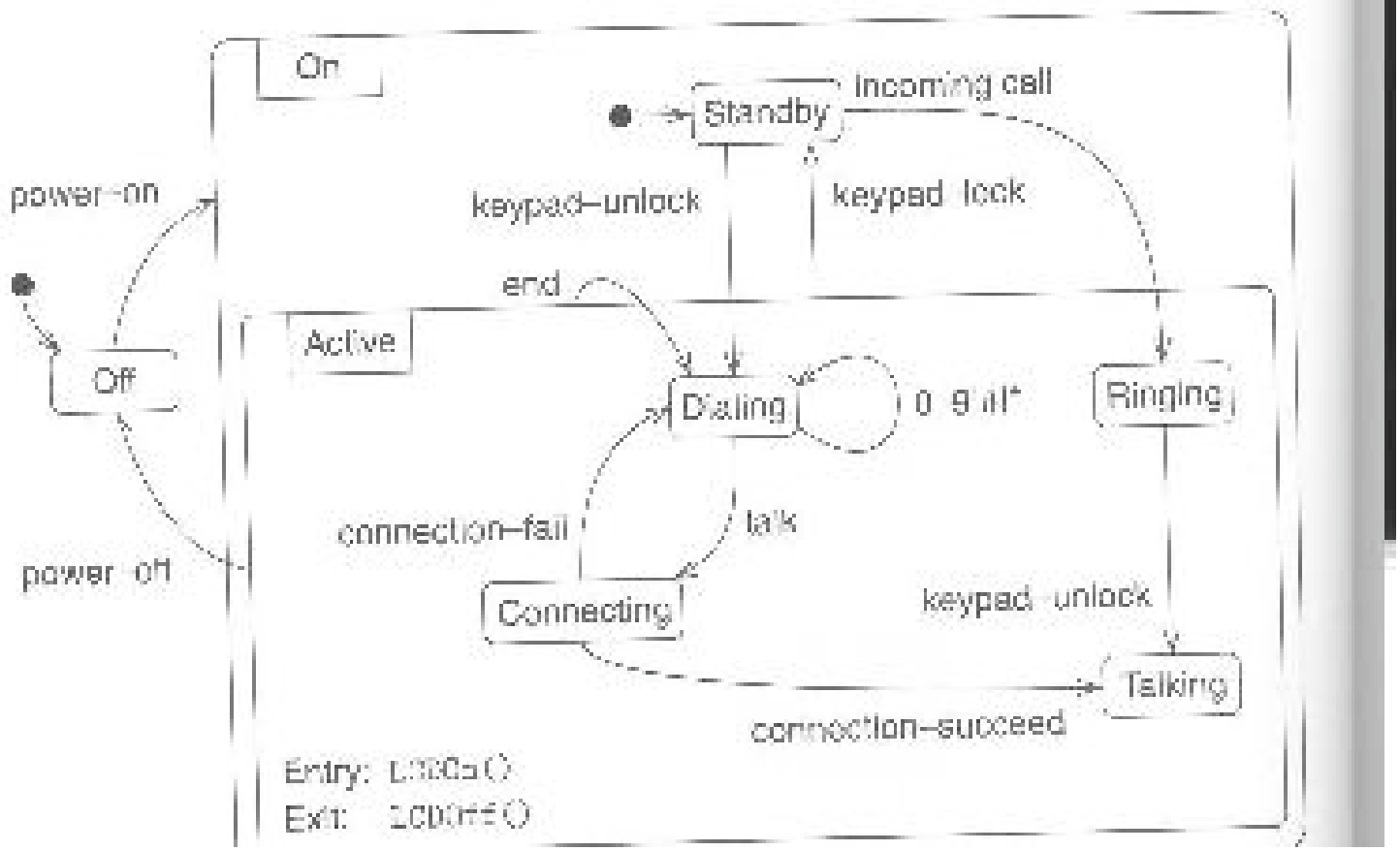


(b) composite concurrent states

Example State Diagram

Figure 2.14

A state diagram: operation of a cellular phone.



Modeling with Use Cases

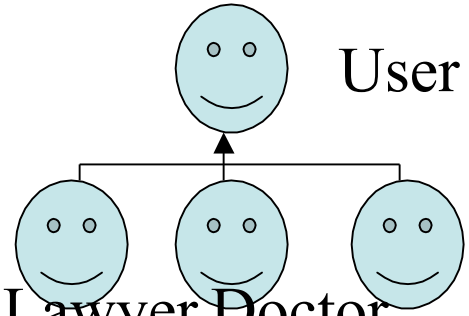
- Use cases and use case diagrams are used to model the requirements of a system to be developed.
- Behavior of the system is described in terms of *actors*.
- Use cases describe what the system does in terms of these actors trying to use the system, and describe how the system does it.
- Users of use cases come up with scenarios that describe in a paragraph a possible flow of events
- Alternatively, one can have two columns: one with actor actions and the other with system responses.

Use Case Diagrams

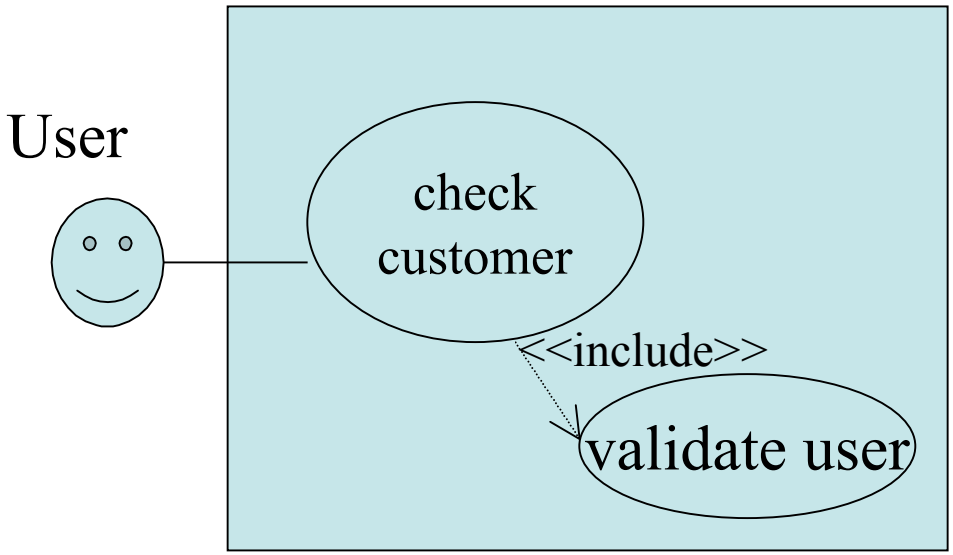
actor

a use case

Can put actors into a hierarchy



Record management system



In addition, can say a use case <<include>> or <<extend>> other use cases.

CRC Cards

- Class Responsibility and Collaborator Cards. (Beck Cunningham OOPSLA ‘89)
- These are another modeling technique like UML modeling, and use case modeling.
- The idea is to work in small teams (5-6) consisting of developers, domain experts, and OO facilitators.
- These teams develop create a sequence of cue cards. Each card contains on it the name of a class, its superclass, its responsibilities, and its collaborators.

More CRC Cards

- Design is broken into sessions where participants try to identify all the nouns and verbs associated with a problems.
- Nouns become the classes and verbs become responsibilities
- Superclasses and collaborators are defined as they become obvious. For instance, if one has several cards with similar responsibilities one defines a superclass with these responsibilities. Collaborators will be classes which are likely to be navigatable to from a given class.