

Use Case Modeling

Dave Levitt

CS 2000:

Systems Analysis & Design

Agenda

- Discuss Article
- Discuss class project
- Introduction to use case modeling
- Managing change
- Advanced Use Case Topics

Intro. To Use Cases

- Topics to cover:
 - Description
 - History
 - Levels of Detail
 - Audience
 - Use Case terminology
 - Steps / Example
 - Relation to UP

Intro. To Use Cases (cont'd)

- Description:
 - Describe a sequence of events that satisfy a user's goal, I.e. provide a useful result to the user.
 - Primarily used to capture functional requirements, but can capture non-functional ones as well.
- History
 - Popularized by Ivar Jacobson, “Object Oriented Software Engineering”
 - Objectory, along with use cases and other works by the 3 amigos (Booch, Jacobson, Rumbaugh) incorporated into UP.
 - Up until a few years ago, there were not many good books on use cases. Today there are several.

Intro. To Use Cases (cont'd)

- Levels of Detail:
 - Summary level (I.e. use case survey, early use case development)
 - Detail level (fully dressed, suitable for testing and other activities)
- Audience
 - Stakeholders, Developers, Testers, Trainers, Architects, UI specialists
 - Get the idea!
- Use Case Terminology
 - Use Case Diagram
 - Use Case
 - Parts of a Use Case (we'll see in the examples)
 - Use Case vs. Use Case Instance / Scenario

Intro. To Use Cases (cont'd)

- Use case instances (a.k.a. scenarios):
 - Capture a specific execution of a use case.
 - One use case can, and usually has several use case instances. Target is 5-7 instances per use case.
 - Useful for:
 - Validating requirements with stakeholders and development team.
 - Starting the testing effort
 - Iteration planning.

Intro. To Use Cases (cont'd)

- Steps
 - Define System Boundary
 - Identify Actors (primary, then secondary / supporting)
 - Identify goals
 - Write use cases
 - Iterate and add details as you learn more

Intro. To Use Cases (cont'd)

- ATM Example
 - Develop Use Case Model
 - Identify Actors
 - Identify Goals
 - Discuss Withdraw Money use case
 - Group Exercise – do the other use cases

Intro. To Use Cases (cont'd)

- Inception Phase:
 - Use Case Model started. Most actors defined.
 - Use cases and supporting artifacts are usually at the summary level. Use cases deemed architecturally significant or otherwise risky may be detailed.
 - Some preliminary UI prototypes are developed.
 - Some analysis and implementation, but not much.
 - Supplementary spec's started.
 - Typically one iteration, perhaps two.

Intro. To Use Cases (cont'd)

- Elaboration Phase:
 - The bulk of the Use Cases and supporting artifacts are captured.
 - Use cases prioritized by risk: technical and business.
 - Use case model refined and updated. More analysis, design, and implementation.
 - Typically more than one iteration.
 - Emphasis is on building an architecture: layers, subsystems, interfaces, implementing / testing some major scenarios.
 - Start usability testing.

Intro. To Use Cases (cont'd)

– Construction Phase:

- Some requirements captured, but hopefully not much.
- Analysis and Design models refined.
- Bulk of the work is on implementation and test.
- Change control becomes critical.

– Transition Phase:

- Requirements, a/d, and implementation taper off.
- Bulk of the work is in deployment.
- Can be a non-trivial task.

Managing Change

- Analysis is a process of discovery.
- Change is inevitable
- If you don't manage change, it will manage you!
- RUP and iterative methods are designed to deal with change. How?
- Relying on RUP alone is not enough.

Managing Change (cont'd)

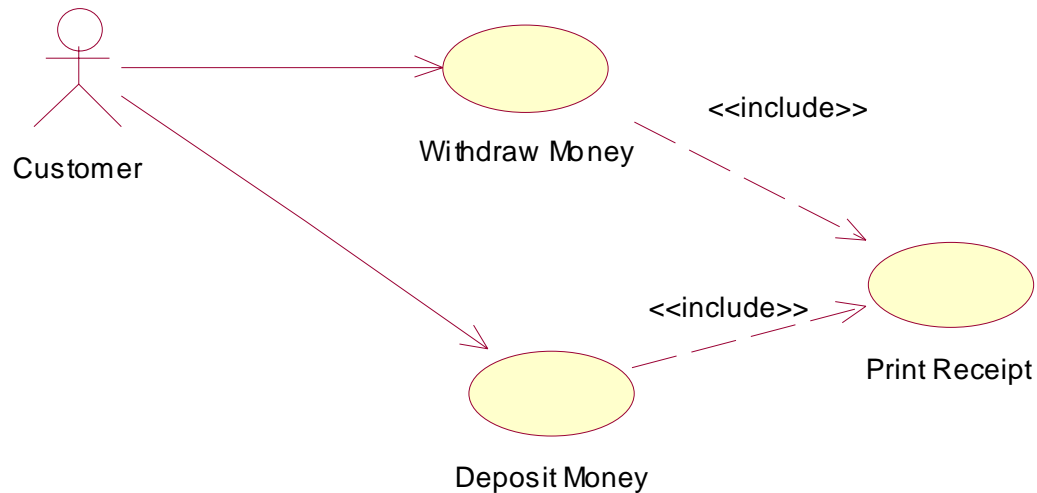
- Concepts:
 - Use Case Approval
 - Baseline
 - Change Request
 - Change Control Board (CCB)

Advanced Use Case Topics

- UML provides other use case modeling relationships:
 - The *include* relationship is used to share functionality between use cases.
 - The *extends* relationship is used when one use case can extend the functionality of another.
 - The *generalization* relationship is used to factor common out common behavior.

Advanced Use Case Topics (cont'd)

Example of an Includes relationship

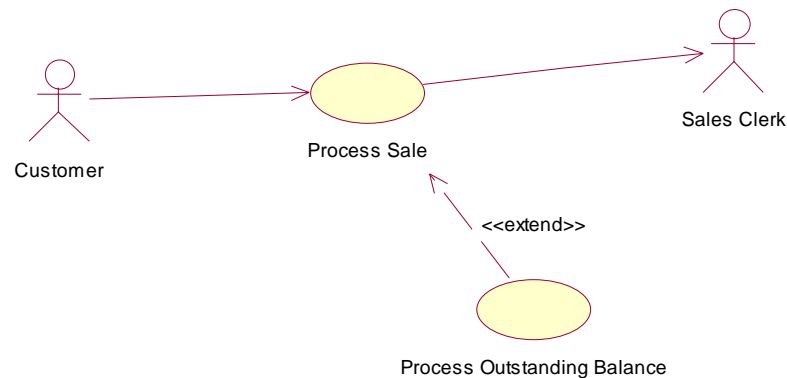


Note the following:

- * The arrow points to the included use case.
- * The relationship is given a stereotype of <<Include>>
- * The entire use case is included - not just part(s) of it.
- * The included use case knows nothing about the use cases that include it.

Advanced Use Case Topics (cont'd)

Example of an Extends relationship

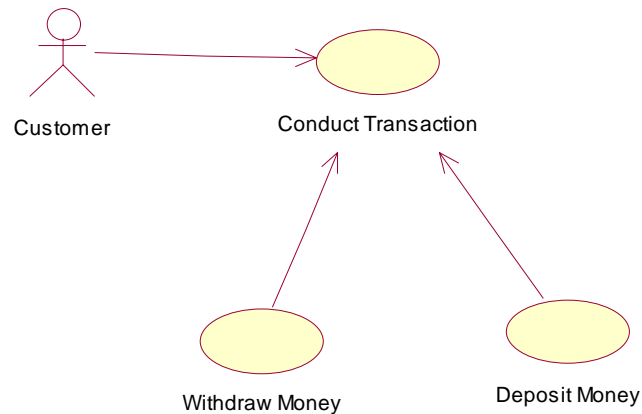


Note the following:

- * The arrow points from the extended use case to the base use case.
 - * The relationship is given a stereotype of <<extends>>
 - * The extended use case can stand on its own.
 - * The extended use case never modifies the base use case.
 - * The extended use case(s) are known as public extension points.
- By contrast, the alternative flows and exception flows within a use case are private flows.

Advanced Use Case Topics (cont'd)

Example of a Generalization relationship

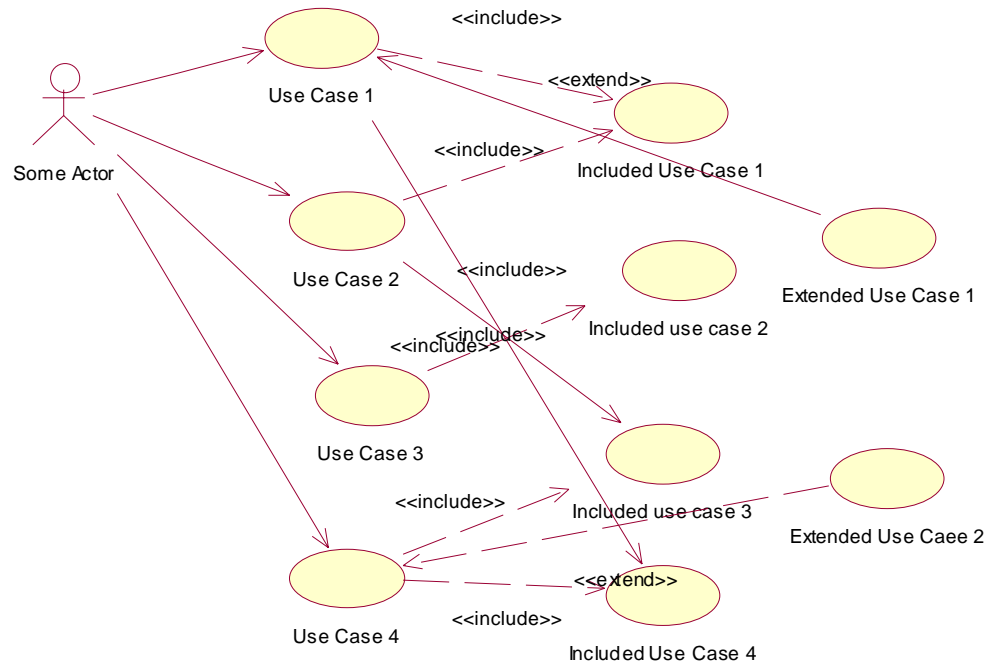


Note the following:

- * The top use case is the most general.
- * The use cases below it denote specializations
- * The specialized use cases are the ones that are performed
- * The specialized use cases use parts (or none of) the generalized use case.

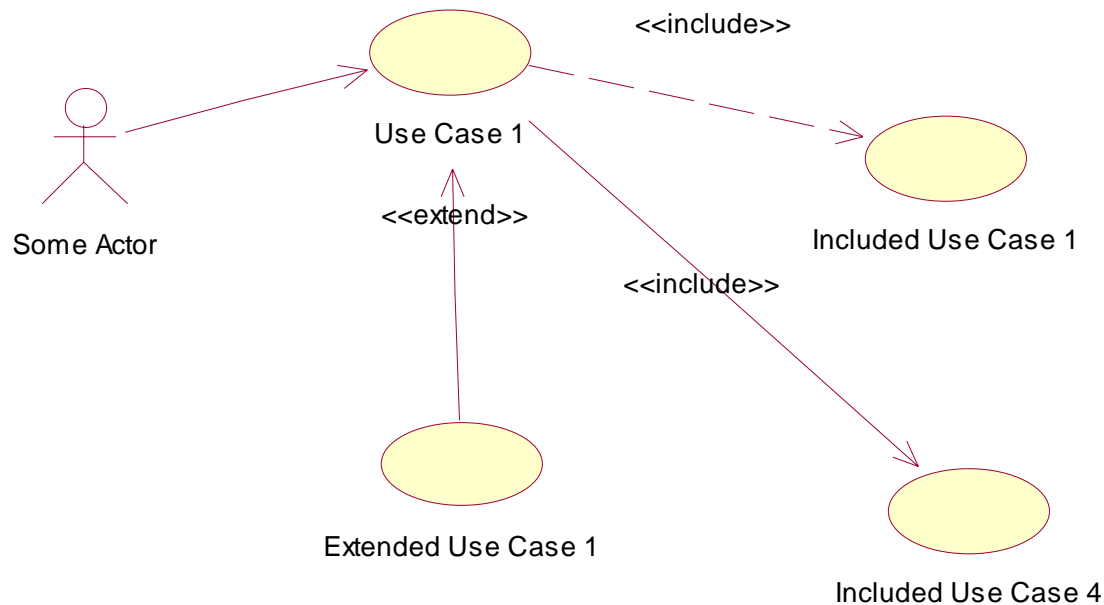
Advanced Use Case Topics (cont'd)

What's wrong with this picture?



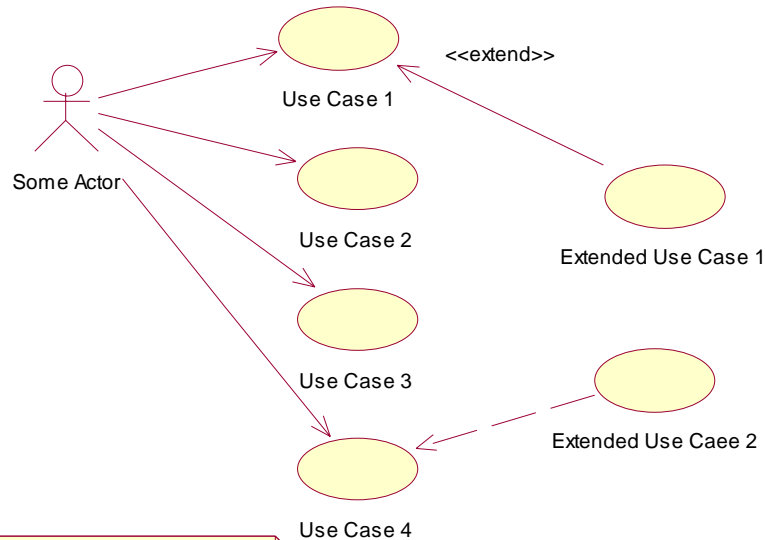
Advanced Use Case Topics (cont'd)

One alternative - factor out use cases



Advanced Use Case Topics (cont'd)

Another Alternative - don't show, but add a note



For purposes of clarity, this diagram does show Included Use Cases 1, 2, 3, and 4. See use case reports for details

Advanced Use Case Topics (cont'd)

- Tenants:
 - Diagrams are an aid to clear thinking. They are a communication tool, so KISS.
 - Do not get carried away with extends, includes, and generalization relationships.
 - They can confuse end users!
 - AND they can confuse modelers too!
 - Focus on writing use cases!