Use Cases

Use cases

Use Cases Diagrams

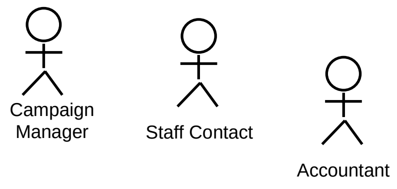
- Textual descriptions of the functionality of the system from user's perspective
  - ✓ In our case we consider is the ACTOR perspective
- Used to show the functionality that the system will provide and which users will communicate with the systems in some way when it provides that functionality
- Developed by I. Jacobson et al
- Part of UML

Actors

- Anything that needs to exchange information with the system
- Anything that is external to the system
- Define roles that users can play
- In our case an ACTOR can be: agent, role or a posisition

Actors

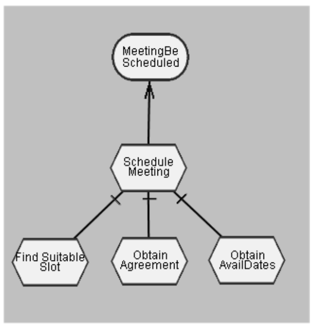
- An actor is someone or some thing that must interact with the system under development



- In our case we can consier as actors also other software systems

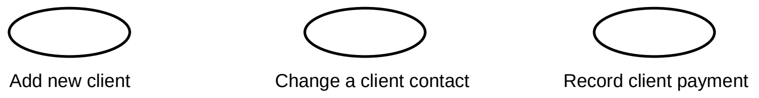
Use Cases

- A use case is a pattern of behavior the system exhibits
  - ✓ Each use case is a sequence of related transactions performed by an actor and the system in a dialogue
  - ✓ In our case we consider the behavior as a particular way to achieve a goal <u>from the user perspective</u>



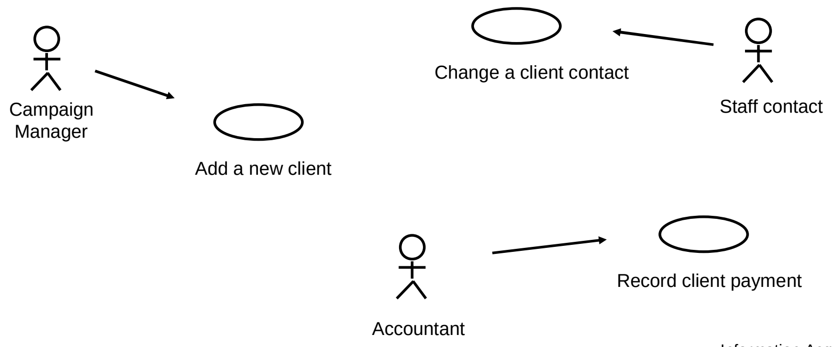
Use Cases

- Actors are examined to determine their needs
  - ✓ Campaign Manager -- add a new client
  - ✓ Staff Contact -- Change a client contact
  - ✓ Accountant -- Record client payment



Use Case Diagram

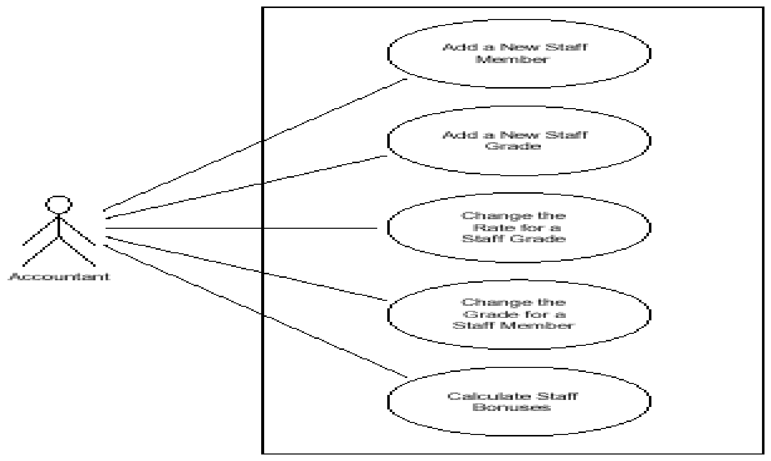
- Use case diagrams are created to visualize the relationships between actors and use cases



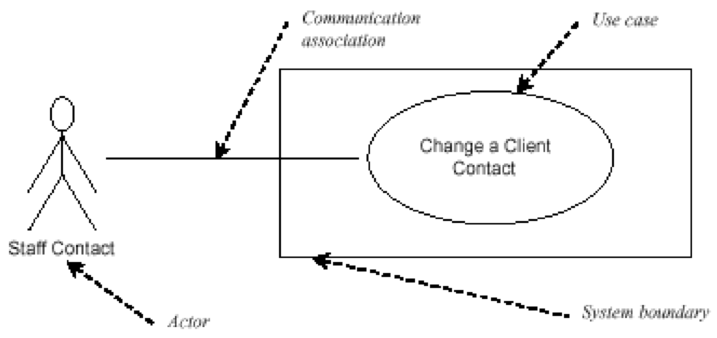
Use Cases

- Purpose
  - ✓ To produce a set of diagrams which summarize the functions which the users expect to find in the system.
  - ✓ To document the scope of the system and the developer's understanding of what it is that users require.
  - ✓ The textual user case descriptions provides a description of the interaction between the users of the system, termed actors, and the high level functions within the system the Use Cases.
- Description
  - ✓ Can be in summary form or in a more detailed form in which the interaction between actor and use case is described in a step-by-step way.
  - ✓ Describes interactions as the user sees it, and it is not a definition of the internal processes within the systems or some kind of program specification.

Agate Case Study

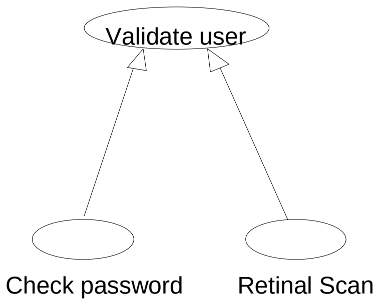


Agate Case Study



Use Cases relationships

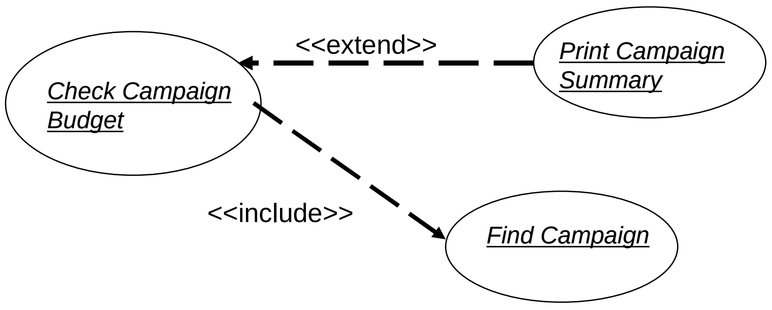
- <<Generalization>>: A relationship between a general use case and a more specific use case that inherits and adds features to it.
- You can find such generalization looking at both SR diagrams and goal models you have produced in the earlier phases.



Use Cases relationships

- **<<**Include**>>:** The insertion of additional behavior into a base use case that explicitly describes the insertion.
  - ✓ Used to avoid describing the same flow of events several times, by putting the common behavior in a use case of its own.
- **<<**Extend **>>:** The insertion of additional behavior into a base use case that does not know about it.
  - ✓ To model a part of a use case the user may see as optional system behavior.
  - ✓ To model a separate subflow that is executed only under given conditions.

Inclusion and Extension



Finding Use Cases

- Ask following questions for each actor
  - ✓Which functions does the actor require from the system? What does the actor need to do ?
  - ✓Does the actor need to read, create, destroy, modify, or store some kinds of information in the system ?
  - ✓Does the actor have to be notified about events in the system? or does the actor need to notify the system about something ? What do those events represent in terms of functionality ?
  - ✓Could the actor′s daily work be simplified or made more efficient through new functions in the system?

Finding Actors

- Can be identified by following questions:
  - ✓Who will use the main functionality of the system(primary actors)?
  - ✓Who will need support from the system to do their daily tasks?
  - ✓Who will need to maintain, administrate, keep the system working(secondary actors)?
  - ✓Which hardware devices does the system need to handle?
  - ✓With which other systems does the system need to interact?
  - ✓Who or what has an interest in the results that the system produce?
- Tips
  - ✓don't only consider the users who directly use the system, but all others who need service from the system

SD, SR and Goal models can help in this

Finding Use Cases

- Without considering current actors
  - ✓What input/output does the system need ? Where does this input/output come from or to go?
  - ✓What are the major problem with the current implementation of this system?

Documenting Use Cases

- A flow of events document is created for each use cases
  - ✓ Written from an actor point of view
- Details what the system must provide to the actor when the use case is executed
- Typical contents
  - ✓ How the use case starts and ends
  - ✓ Normal flow of events
  - ✓ Alternate flow of events
  - ✓ Exceptional flow of events