VIII. Use Cases

The Unified Modeling Language
Actors and Use Cases
How to Find Them

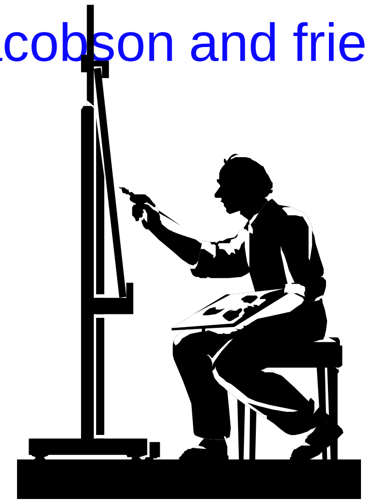
Acknowledgment: these slides are based on Prof. John Mylopoulos slides which are used to teach a similar course in the University of Toronto – St. George campus. Used with Permission.

The Unified Modeling Language (UML)

- Booch and Rumbaugh started working towards a unified modelling language (UML) in 1994 under the auspices of Rational Inc. They were later joined by Jacobson.
- UML only offers a notation, not a methodology for modeling (as various OOA techniques do).
- Combines Jacobson’s use cases with Booch and Rumbaugh concepts for object modeling, along with statecharts.
- UML has been adopted by the Object Management Group {OMG) as an (object) modelling standard. OMG UML 1.0 is the first version of this new modelling standard.

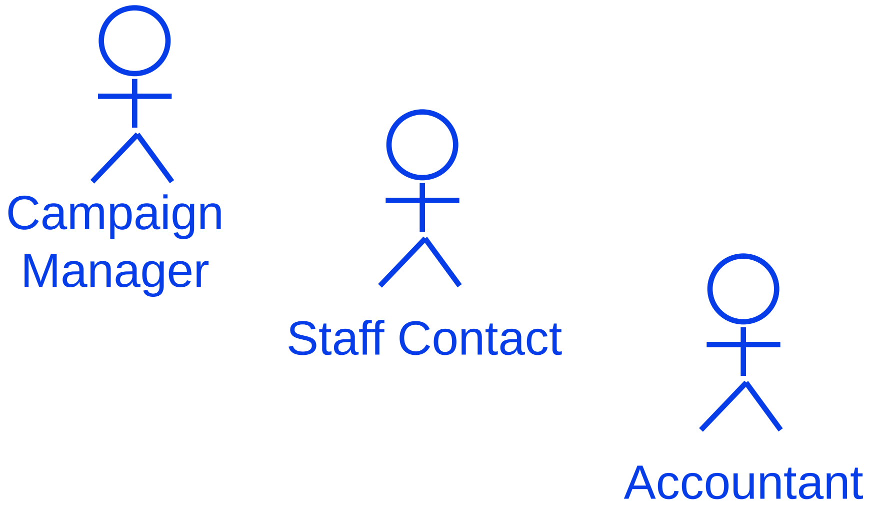
Where Do We Start? Use Cases

- Use cases are descriptions of the functionality of the new system (or any artifact under design, for that matter!) from a user’s perspective.
- They answer the question: How will the artifact be used, once it is built?
- Used to show the functions to be provided by the artifact, also which users will use which functions.
- Developed by Ivar Jacobson and friends [Jacobson92].



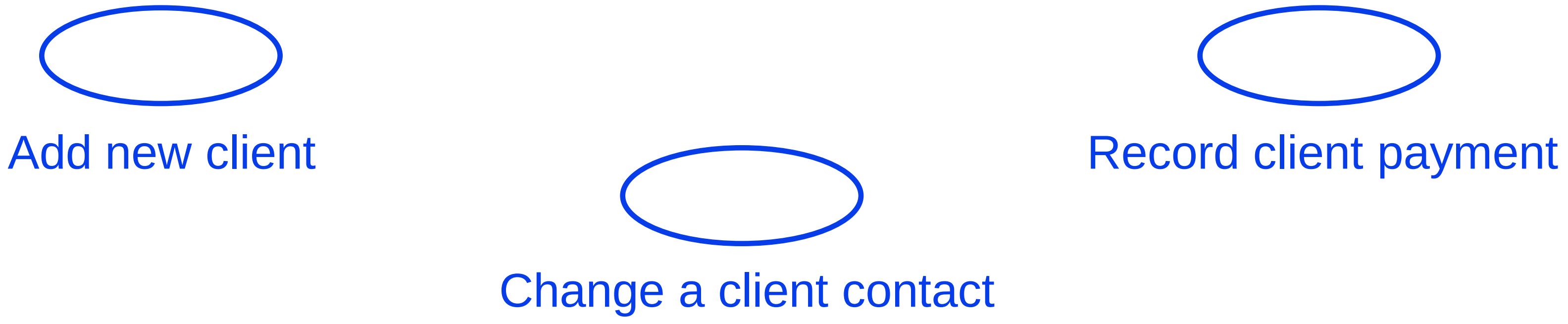
Actors

- An actor is anything that needs to exchange information with the artifact
- An actor could be a person, or another external, system.
- Actors define ***roles*** that users can play while using the artifact.



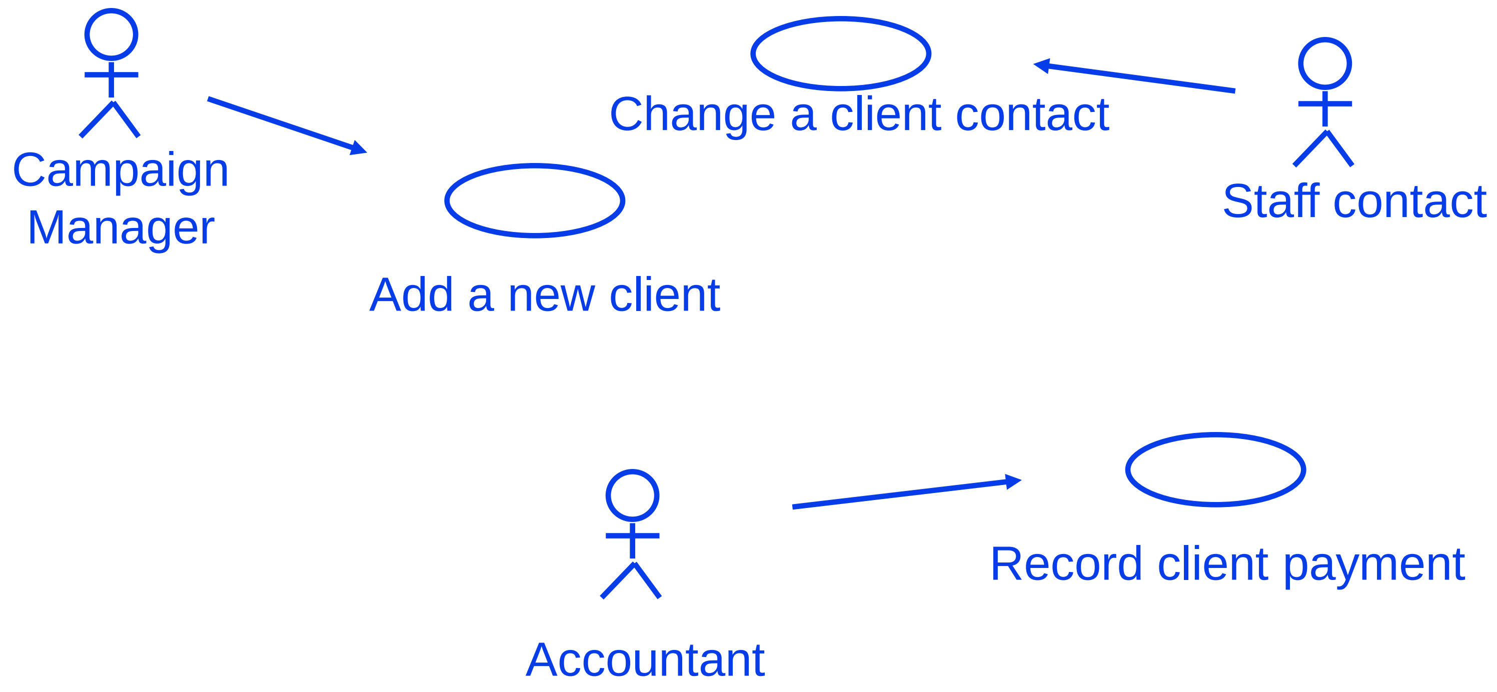
Use Cases

- A ***use case*** is a pattern of behavior which the new system is required to exhibit.
- Each use case is a sequence of related transactions performed by an actor and the system through a dialogue.
- To find use case, examine each actor and her needs, e.g.,
  - ✓ Campaign Manager -- add a new client
  - ✓ Staff Contact -- Change a client contact
  - ✓ Accountant -- Record client payment

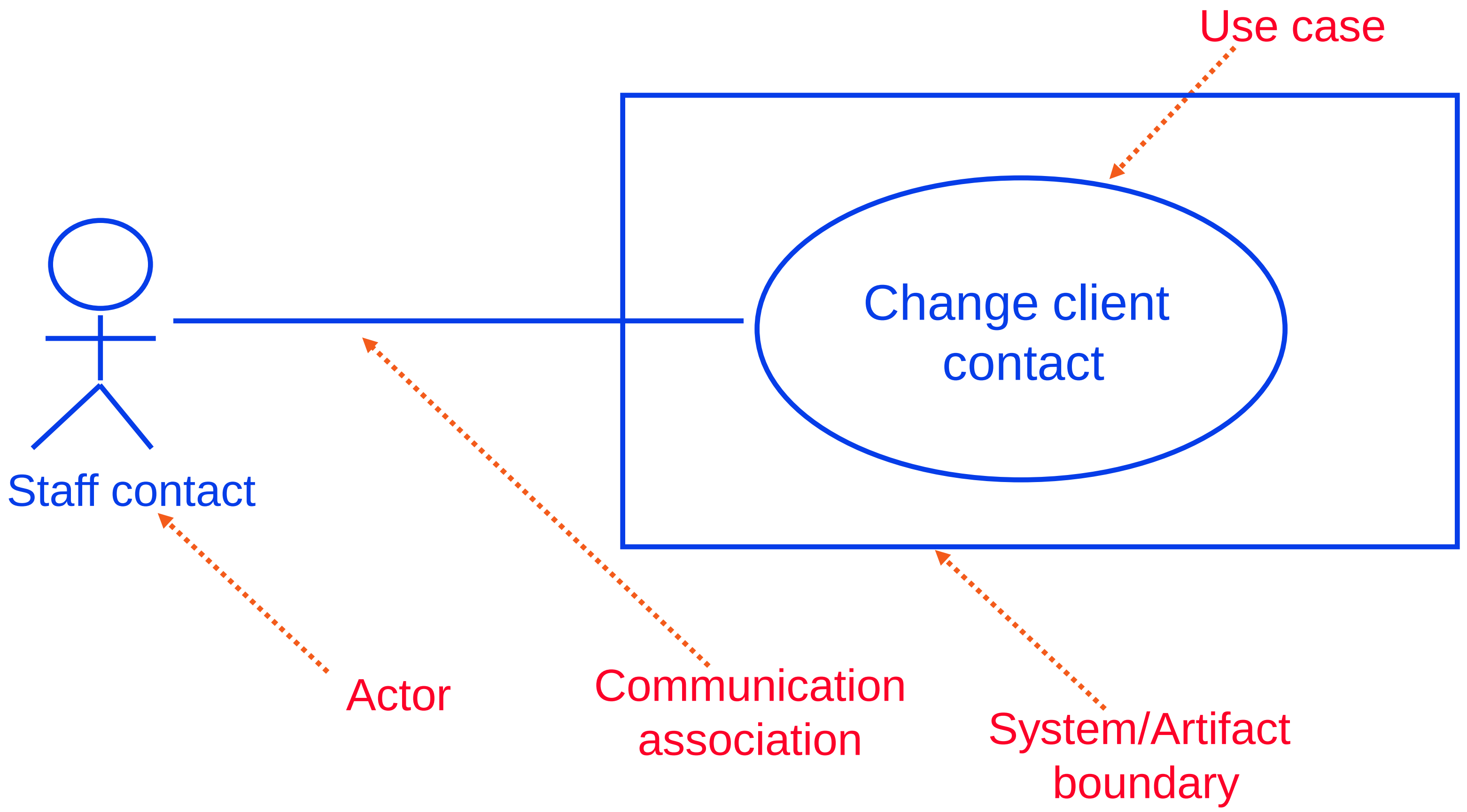


Use Case Diagrams

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



Notation for Use Cases



Agate is an Advertising Company

…which puts together advertising campaigns for client companies. Here is the breakdown of their staff:

Direction
1 Campaign
1 Creative
1 Admin
1 Finance

Admin
1 Office mgr
3 Direction asst
4 Manager clerks
2 Receptionists
2 Clerks/typists

Campaigns Mgt
2 Campaign managers
3 Campaign marketers
1 Editor in Chief
1 Creative Manager

Edition
1 Filing clerk
2 Editors
4 Copy writers

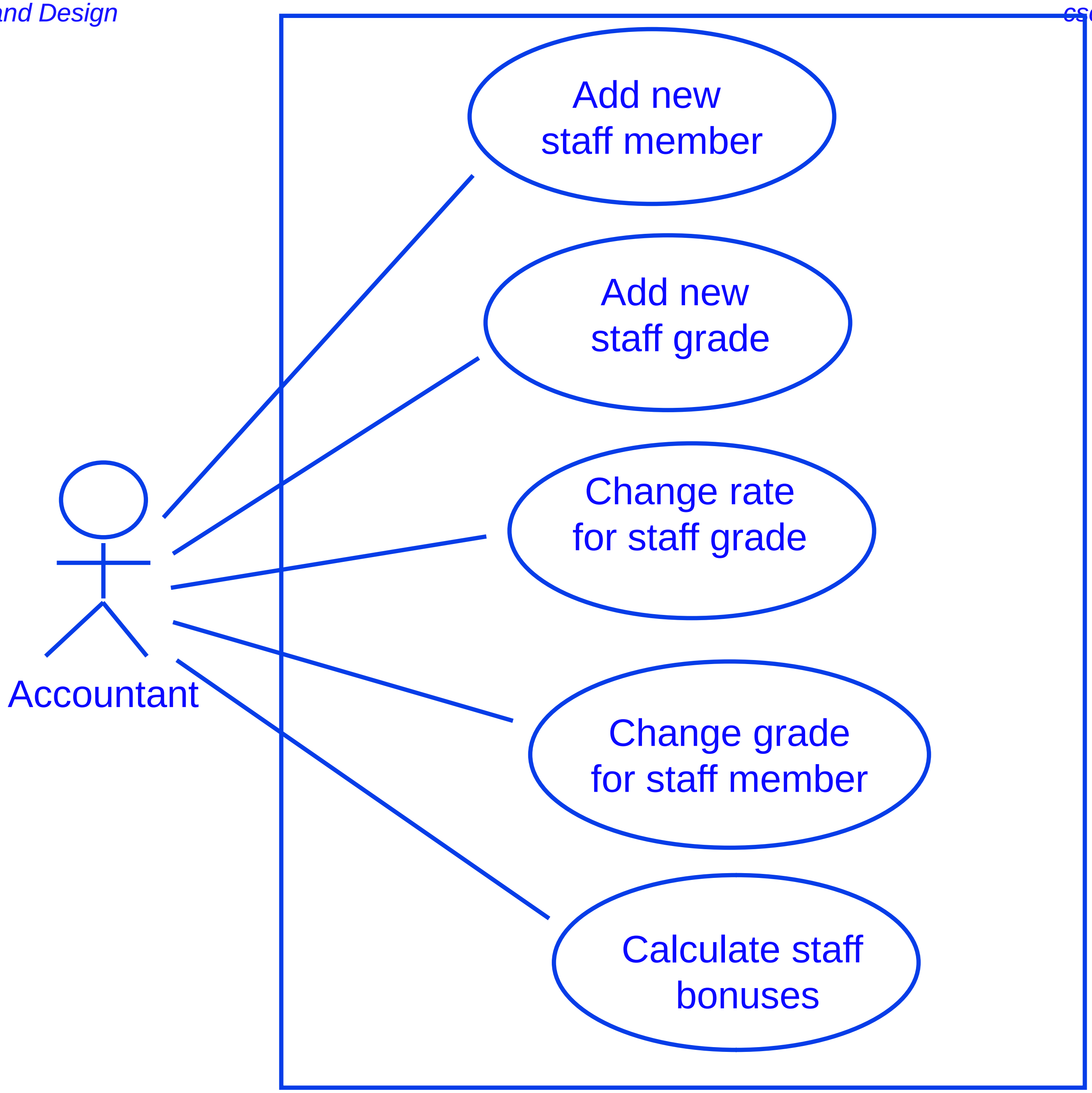
Graphics
6 Graphic designers
2 Photographers

IT
1 IT manager
1 Network administrator
1 System admin
1 Analyst

Accounts Edition
1 Accountant manager
1 Credit controller
2 Accounts clerks
2 Purchasing assistants

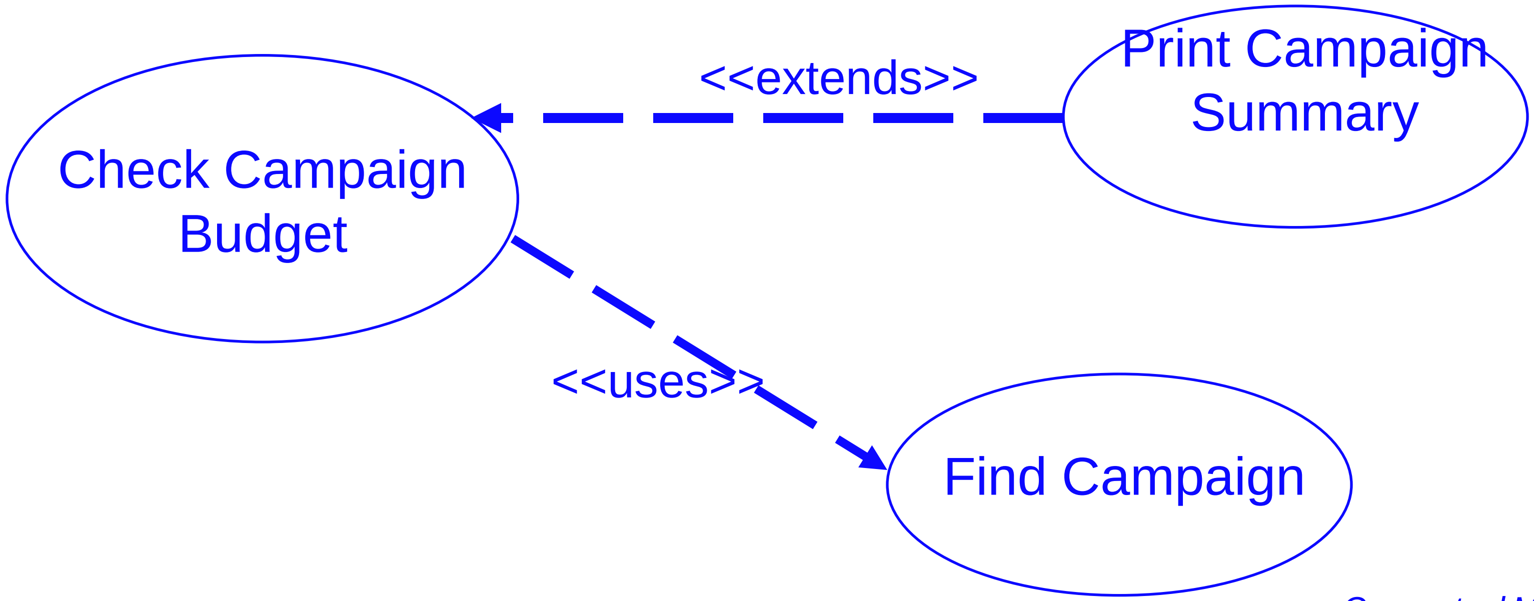
Documentation
1 Media librarian
1 Resource libr
1 Knowledge worker
1 Computer tech

Agate Case Study



<<extends>> and <<uses>>

- <<extends>> implies that one use case adds behaviour to a base case; used to model a part of a use case that the user may see as optional system behavior; also models a separate sub-case which is executed conditionally.
- <<uses>>: adds behavior to a base case (like a procedure call); used to avoid describing the same flow of events several times, by putting the common behavior in a use case of its own.



Finding Actors

- Actors can be identified by answering the following questions:
  - ✓ Who will be a primary user of the artifact? (primary actor)
  - ✓ Who will need support from the artifact to do her daily tasks?
  - ✓ Who will maintain, administrate, keep the artifact working? (secondary actor)
  - ✓ Which hardware or other devices does the system need?
  - ✓ With which other systems does the artifact need to interact with?
  - ✓ Who or what has an interest in the results that the artifact produces ?
- Tip: don’t consider only the users who directly use the artifact, but also others who need services from the artifact!

Finding Use Cases

For each actor, ask the following questions:

- Which functions does the actor require from the artifact? What does the actor need to do?
- Does the actor need to read, create, destroy, modify, or store some kinds of information in the artifact?
- Does the actor have to be notified about events in the artifact? Or, does the actor need to notify the artifact about something? What do those events require in terms of artifact functionality?
- Could the actor's daily work be simplified or made more efficient through new functions provided by the artifact?

Documenting Use Cases

- For each use case, prepare a “flow of events” document, written from an actor’s point of view.
- The document 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;