Chapter 3

Object Oriented Analysis

Prof. Mirco Tribastone

29.11.2012

Goals

- To carry out a use case analysis to a developed system.
- Step-by-step creation of a corresponding static model (e.g., a class diagram).
- Creation of interaction diagrams for communicating objects.
- Deduction of state and activity diagrams (from an interaction model).

The presented method is based on

- OMT (Rumbaugh et al.)
- "Uses Cases" from Objekt Orientierte Software Entwicklung (Jacobson et al.)

Goal: Precise and plain description of requirements.

We achieve this by:

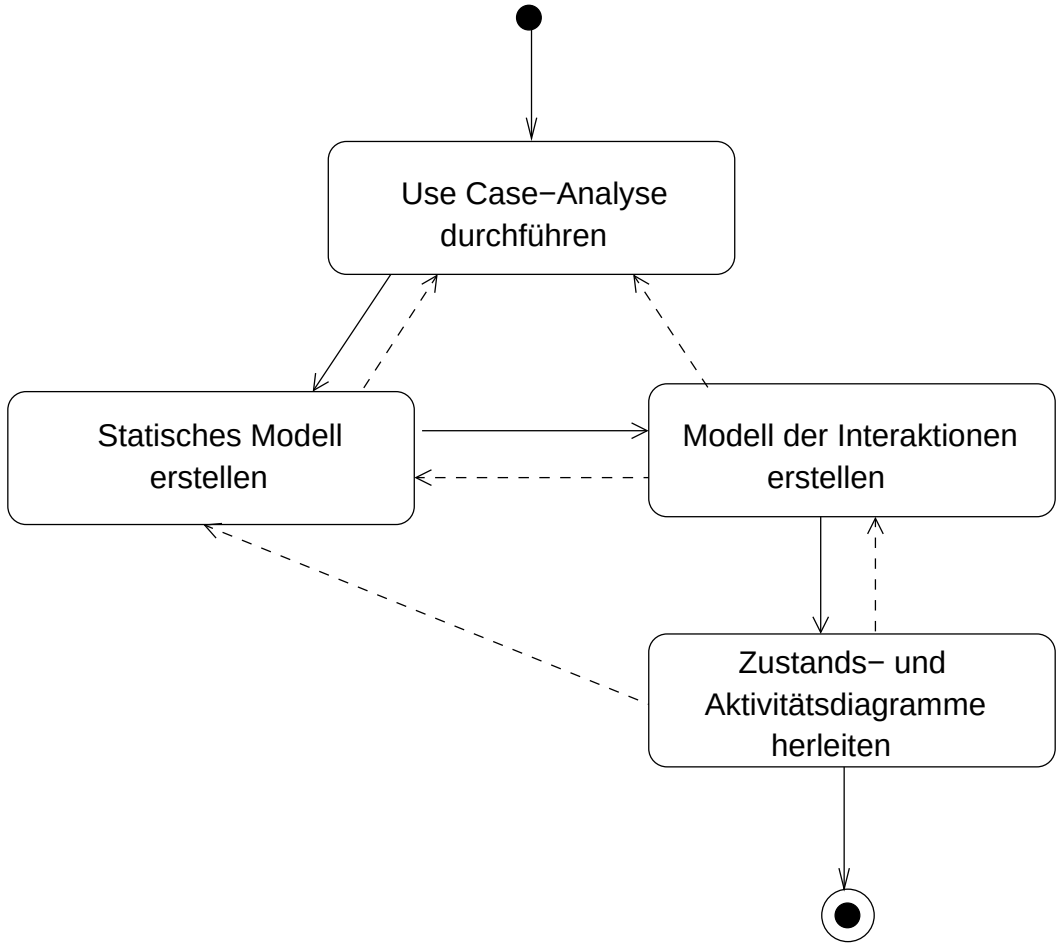
1. "use case"-analysis
2. developing of a static model, given by a class diagram
3. developing of a dynamic model, given by
   (a) interaction diagrams
   (b) state and activity diagrams

Remarks

- The above steps will be supplemented by validating, reengineering and extending of the corresponding models (in several iterations).
- We can also
  - build a coarse architecture of the system
  - sketch the user interfaces

  during the analysis phase.

Process model of the object orientated analysis



3.1 Use Case Analysis

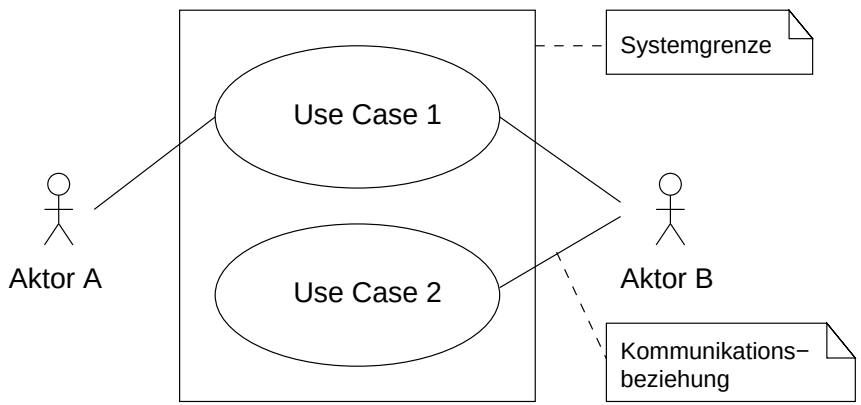
Starting point: informal and short problem description.

Goal: describing the functionality of the desired system.

3.1.1 Use Case Model

- Consists of *actors* and *use cases*.
- Describes an external view on the system.

Forms of representation:



Actors

- Exchange information with the system from the outside (users, other systems, devices).
- Actors are characterised by their *role* with respect to the system.

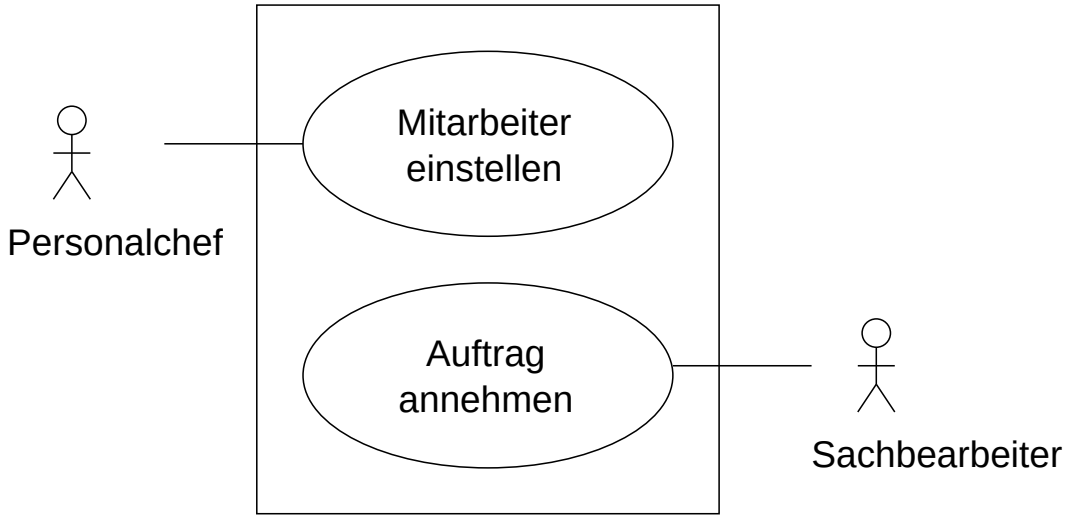
Use case

- Describes functional requirements of the system.
- Describes the interactions between one (or several) actors and the system during the processing of a certain task.

Definition (Jacobson):

A use case is a set of cohesive transaction sequences which are executed by a system and lead to a recognisable result.

Example:



Note:

Use cases are usually the computer aided parts of a business process.

3.1.2 Creation of a use case

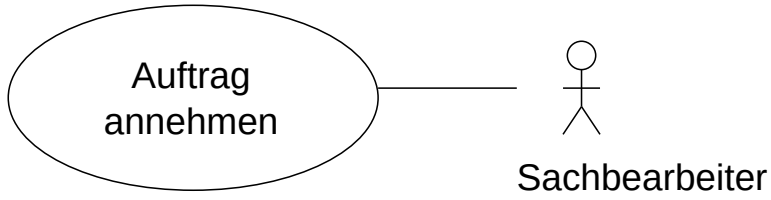
1. Determine the *actors* which interact with the system.
   (Who uses the system? Who gets/puts information from/into the system?)
2. Determine the *use cases* from the tasks the system has to perform for each actor.
   Subtle point: choosing the right granularity.
3. Create a use case diagram (perhaps with a brief description of actors and use cases).
4. Describe the use cases (iterative).

A *use case description* consists of:

- name of the use case
- short description
- preconditions
  (condition for a successful execution of a use case)
- postcondition
  (state after a successful execution)
- a standard run (primary scenario)
  (steps or interactions that would occur during a normal execution of the use case)
- several alternative runs (secondary scenarios)
  (in case of errors and different options)

One can additionally give an activity diagram for the use case.

Example:



Use case
Accept of order

Short description
A case handler accepts the order from a client.

Precondition:
The system is ready to accept an order.

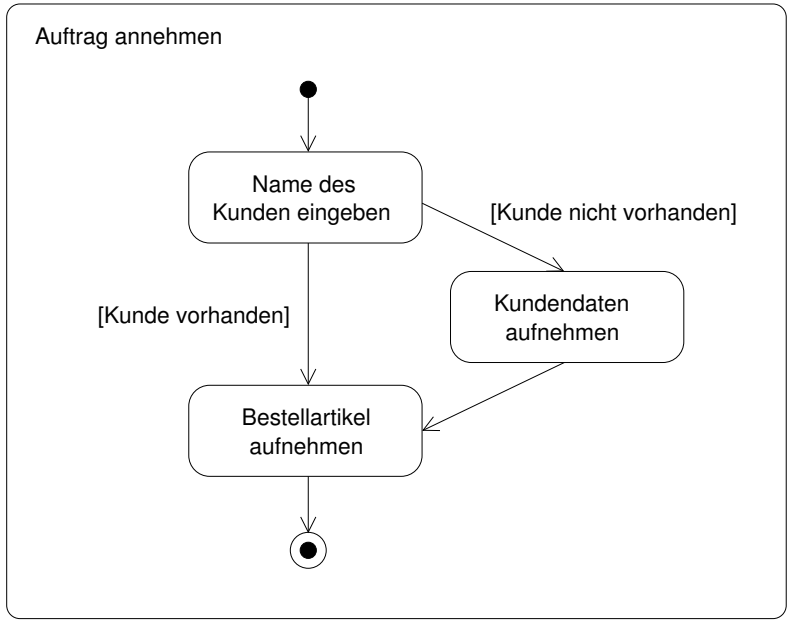
Postcondition:
The order was accepted.

Primary scenario:

1. The clerk inputs the name of the client.
2. The system displays the client data.
3. The clerk inputs the data of the ordered article.

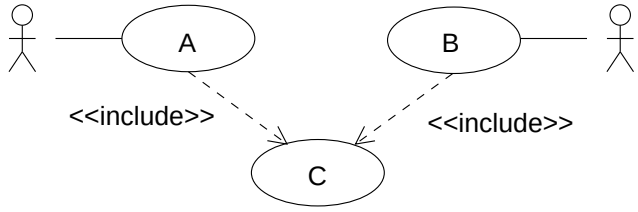
Secondary scenario:
Client not in database

Activity diagram:



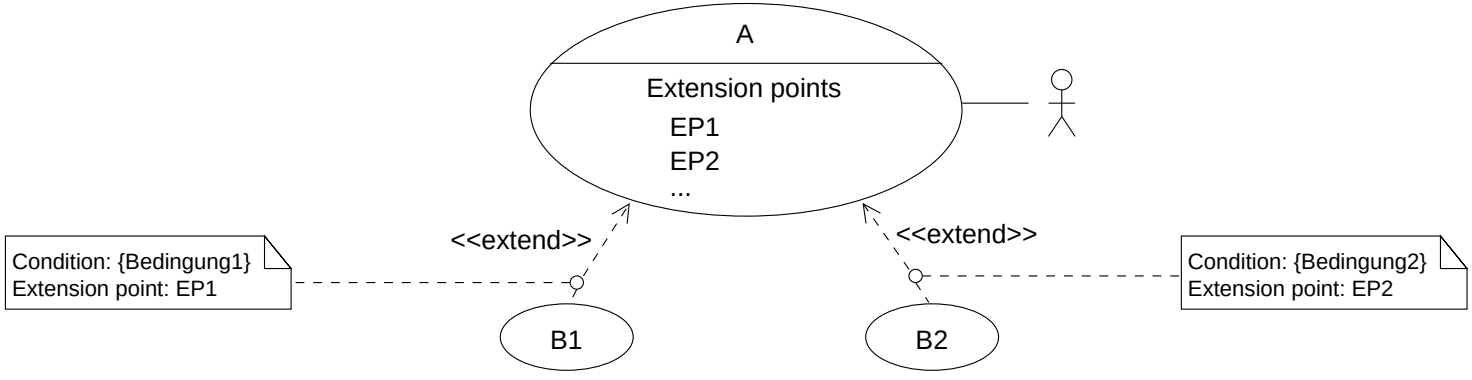
3.1.3 Relations between use cases

1. *Contains-Relation*



Each sequence of A (resp. B) contains (always) as a subsequence a sequence of C.

2. *Extension-Relation*



Extends the behaviour of A if a condition is fulfilled.

3.1.4 Example ATM (Automatic Teller Machine)

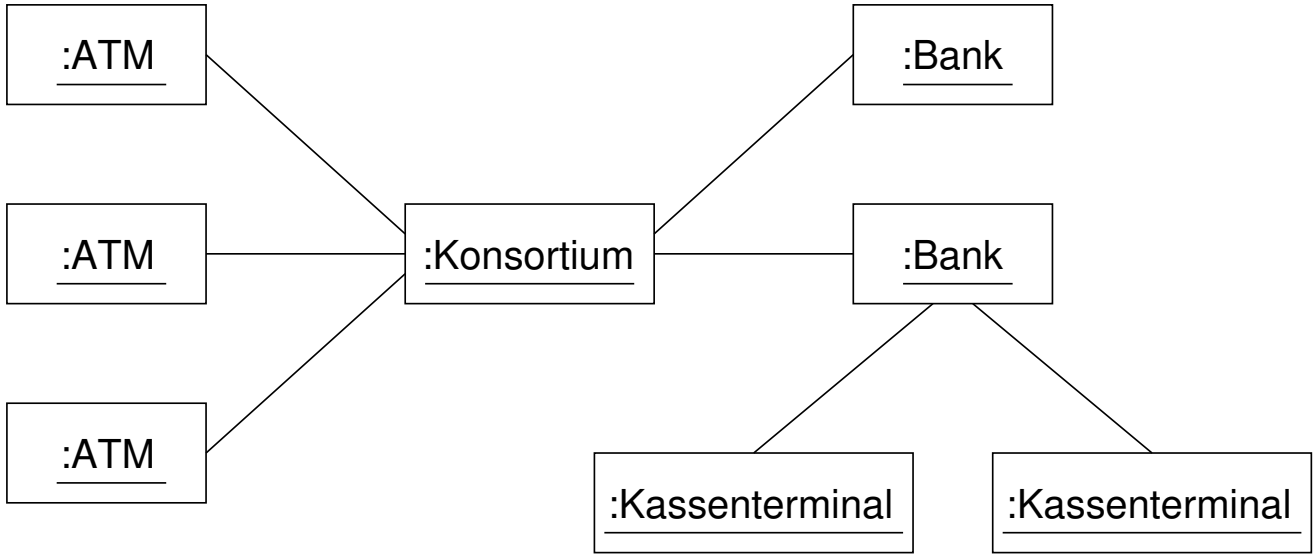
A Network of ATMs
(aus Rumbaugh et al., 1993)

Entwickelt werden soll Software zur Unterstützung eines rechnergesteuerten Bankennetzwerks einschließlich Kassierern und Bankautomaten (ATMs), das sich ein Bankenkonsortium teilt. Jede Bank besitzt einen eigenen Computer, auf dem sie ihre Konten verwaltet und die Transaktionen auf Konten durchführt. Die Banken besitzen Kassenterminals, die direkt mit dem bankeigenen Computer kommunizieren.

Kassierer geben Konto- und Transaktionsdaten ein. ATMs kommunizieren mit einem Zentralrechner, der Transaktionen mit den jeweiligen Banken abklärt. Ein ATM akzeptiert eine Scheckkarte, interagiert mit dem Benutzer, kommuniziert mit dem zentralen System, um die Transaktion auszuführen, gibt Bargeld aus und druckt Belege.

Das System erfordert geeignete Aufzeichnungsmöglichkeiten und Sicherheitsmaßnahmen. Das System muss parallele Zugriffe auf das gleiche Konto korrekt abwickeln. Die Banken stellen die SW für ihre eigenen Computer selbst bereit.

We have to implement the software for the ATM network.

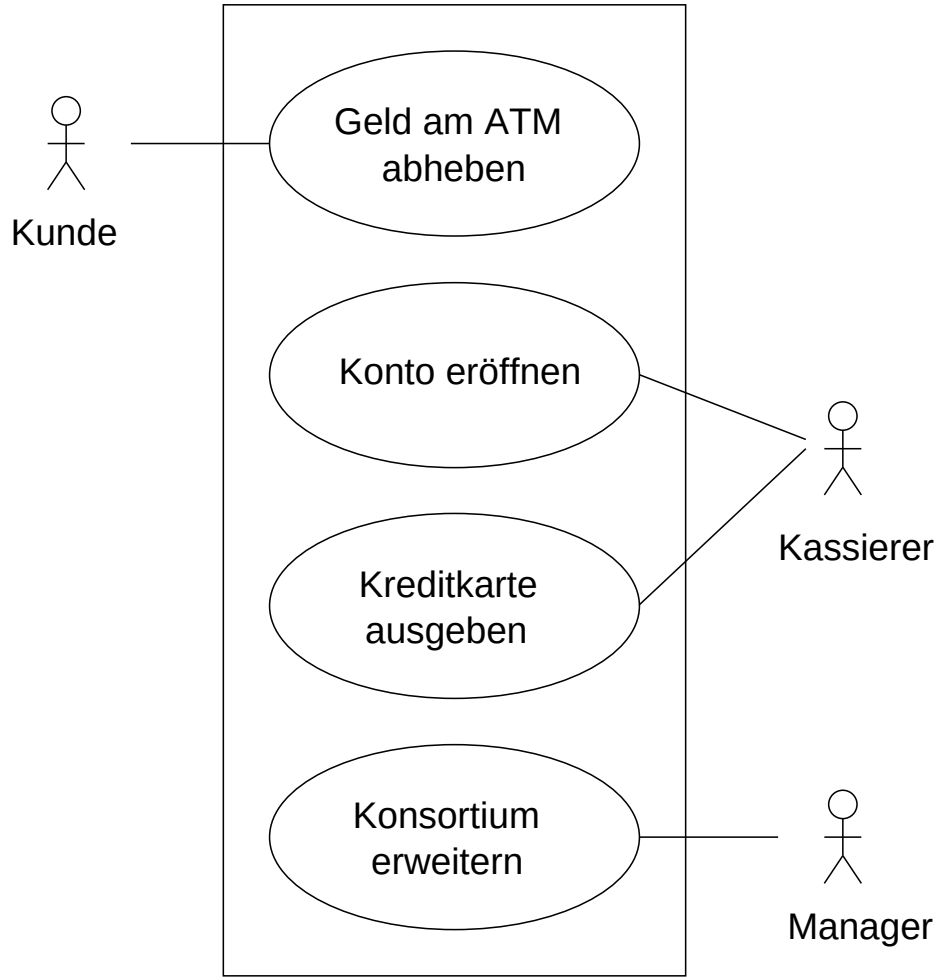


Main Informaion

1. *Actors:* client, teller, manager, ...
2. *Use Cases:*
   - Withdraw money at ATM:
     A client withdraws money from his bank account at an ATM.
   - Open an account:
     A teller creates a new bank account for a client.
   - Get new credit card:
     A teller gives a new credit card to client.
   - Extend consortium:
     A manager from the consortium adds a new bank to the consortium.

   and so on (further use cases can be "deposit money", "make transfer").

3. *Use Case Diagramm* (snippet):



4. *Use Case Description:*

Use case:
Withdraw money at ATM.

Short description:
A client withdraws money from his bank account at an ATM.

Precondition:
ATM is ready to perform a transaction for the client.

Postcondition:
The client has his/her credit card, the money and the receipt.
ATM can serve the next client.

Primary scenario:

1. The client inserts his credit card.
2. The ATM reads the credit card and asks for the PIN.
3. The client enters the PIN.
4. The ATM reads the PIN and checks it. Then, it delegates bank code and credit card number to the consortium for further checking.
5. The consortium checks the bank code and credit card number and responds with OK.
6. The ATM asks the client for his choice (Draw, Deposit, Transaction, Statement).
7. The client chooses "Draw".
8. The ATM asks for the amount of money.
9. The user enters the desired amount of money.
10. The ATM reads the amount of money, checks whether it is within bounds and delegates then the transaction task to the consortium.
11. The consortium delegates the request to the bank of the client. The bank updates the balance and confirms the execution.
12. The consortium informs the ATM about the successful completion of the request.
13. The ATM gives the desired amount of money to the client.

14. After the client takes the money, the ATM ask him/her whether he/she intends to do another transaction.
15. The client says no.
16. The ATM prints a receipt and gives the credit card to the client.
17. The client takes the receipt and the credit card.
18. The ATM asks for the credit card (of the next client).

Secondary scenarios:

Credit card blocked
The bank observes in step 5 that the credit card is blocked and informs the consortium. The consortium delegates this message to the ATM. The ATM informs the client and enters the step 16.

Transaction fails
The bank observes in step 10 that the amount of money is too large and informs the consortium about the failure of the transaction. The consortium delegates this to the ATM. The ATM delegates this to the client. The use case enters then the step 6.

User exits

Credit card not readable

Wrong PIN

Wrong bank code

Not enough money in the ATM

Network down

⋮

Conclusion of Section 3.1

- A use case model consist of use cases and actors.
- An actor exchanges information with the system.
- A use case describes a task which is performed by the system in cooperation with one (or several) actors.
- A use case description contains one primary and several secondary scenarios.
- Use cases can be structured by <<include>> and <<extend>> relations.
- A use case model is build in several steps.