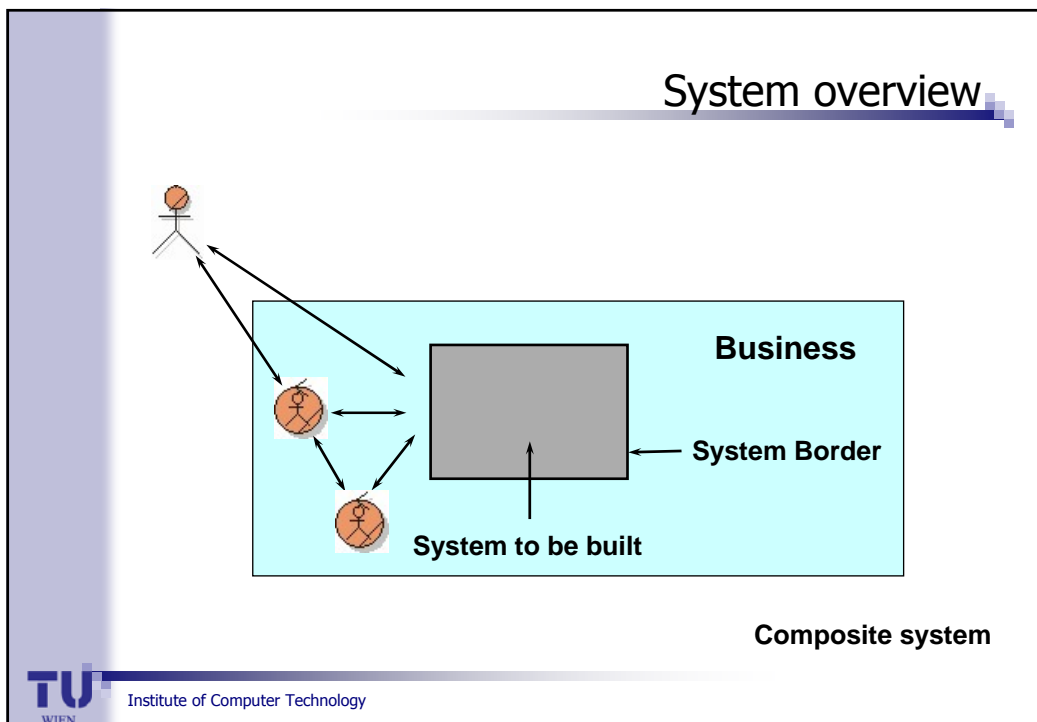


TU
WIEN

Bridging Requirements, Use Cases and Object-Oriented Modeling for Systems Engineering

Institut für
Computertechnik
ICT
Institute of
Computer Technology

Hermann Kaindl
Vienna University of Technology, ICT
Austria




Outline

- Background
- Functional requirements, goals and scenarios / use cases
- Requirements and object-oriented models
- Systematic process
- Final discussion



Institute of Computer Technology

What are requirements?

- User wishes / needs 
- *IEEE Standard:*
"A condition or capacity needed by a user to solve a problem or achieve an objective."
- "The <system> shall be able to ..."
 - system to be built
 - composite system
- *Example:* "The ATM shall accept a cash card."
- Requirements modeling



Institute of Computer Technology

What are requirements? – In practice

- User requirements documents
- Software/system requirements documents
- Mostly descriptions in **natural language**
- Representation often unstructured
- Ad hoc process
- Communication problem
- Requirements and **use cases?**



Institute of Computer Technology

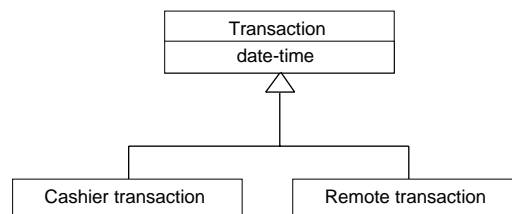


Class and generalization

Class in **UML** (Unified Modeling Language)
<http://www.omg.org>

Generalization / Specialization
(in UML notation)

Object – **instance**



Institute of Computer Technology

Inheritance

Example: **attribute** date-time

Mechanism for information sharing

- Structure (variables, attributes)
- Behavior (methods, procedures)

Multiple inheritance

various theories

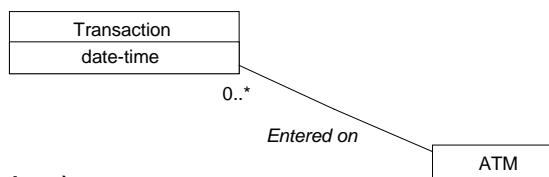


Institute of Computer Technology



Attributes and associations

Attribute for representation of property



Association
(in UML notation)

Relation for linking instances

Multiplicity: range of allowable cardinalities



Institute of Computer Technology

Methods and messages

Methods are procedures / functions

Interface protocol for indirect calls

Sending and receiving of **messages**

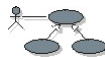
Actual method to be invoked determined through rules for processing a message, e.g., "dynamic binding"



Institute of Computer Technology

Use cases

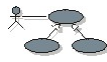
- "particular cases of how the system is to be used"
- Use-Case Report (according to Unified Process):
 1. Brief Description
 2. Flow of Events
 3. Special Requirements
 4. Pre-conditions
 5. Post-conditions
 6. Extension Points
 7. Relationships
 8. Use-Case Diagrams
 9. Other Diagrams



Institute of Computer Technology


Use-case diagram

- UML graphical notation
- Ellipse: use case



Name of use case

- Stick man: actor

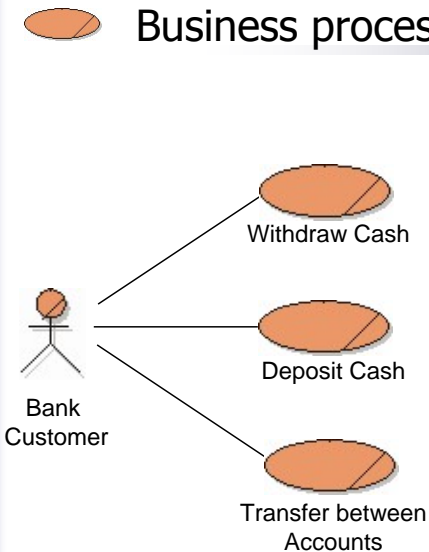





Name of actor

- Connecting line: association

TU WIEN Institute of Computer Technology

Business process — Business Use Case



- Focus on "use" of business by **business actor** 
- Behavior realized by **business workers**  (active) and **business entities**  (passive)
- Special notation in RUP® (Rational Unified Process®)

TU WIEN Institute of Computer Technology

Iterative and incremental development

- **Iterative** development:
Repetition in the process
- **Incremental** development:
Extension of the software in (small) portions



Institute of Computer Technology

Iterative and incremental dev. – Advantages

- Unstable or unclear requirements better manageable
- Continuous integration
- Running version “anytime”
- Smaller risk with novel products
- Reduction of error costs (“fail fast”)



Institute of Computer Technology

Iterative and incremental dev. – Issues

- Stable architecture needed
- Concrete vision for development needed
- Dynamic project management needed
- Commissioning of projects more difficult



Institute of Computer Technology

Outline

- Background
- ■ Functional requirements, goals and scenarios / use cases
- Requirements and object-oriented models
- Systematic process
- Final discussion



Institute of Computer Technology

Glossary

Functions: "effects achieved by some entity"

Goals: "partially specified states that the user considers as desirable"

Scenarios: "sequences of actions aimed at accomplishing some task goal"

Use cases: "particular cases of how the system is to be used", "classes of scenarios"



Institute of Computer Technology

Functional requirements

- Describe required functionality not yet available
- Functional user requirements may be high-level statements of what the system should be able to do.
- Functional software/system requirements should describe the functions of the software/system to be built in detail (but not yet its design or implementation).



Institute of Computer Technology

Scenarios – Stories and narratives

- For representation of
 - cultural heritage
 - explanations of events
 - everyday knowledge
- Human understanding in terms of specific situations
- Human verbal interactions by exchanging stories



Institute of Computer Technology

Scenarios – Video Store example

Rent Available Video:

1. A member of the video store identifies himself/herself to VSS (Video Store Software).
2. VSS shall check the identification.
3. If the identification is successful, VSS shall start a transaction and present a selection of video titles.
4. The member selects a video title that is available and indicates the intent to rent (a copy of) the video.



Institute of Computer Technology

Scenario – Video Store example (cont.)

5. VSS shall book this rental on the account of the member and ask the clerk to hand out a video copy to the member.
6. The clerk hands out a copy of the video title and acknowledges this to VSS.
7. VSS shall again present a selection of video titles.
8. The member does not select further titles, but initiates the termination of the transaction.
9. VSS shall issue a good-bye message and terminate the transaction.



Institute of Computer Technology

By-Function – Video Store example

1. A member of the video store identifies himself/herself to VSS (Video Store Software).

2. VSS shall check the identification.

By-Function: Member Identification Check

...

5. VSS shall book this rental on the account of the member and ask the clerk to hand out a video copy to the member.

By-Function: Video Rental Booking
Video Handing-out Request

...



Institute of Computer Technology

Functional requ. – Video Store example

Rent Available Video *By-Function* *Video Rental Booking*

Video Rental Booking:

VSS shall book the rental of a copy of a video title on the account of the member, and reduce the number of available copies of the video title by 1.



Institute of Computer Technology

Goal – Video Store example

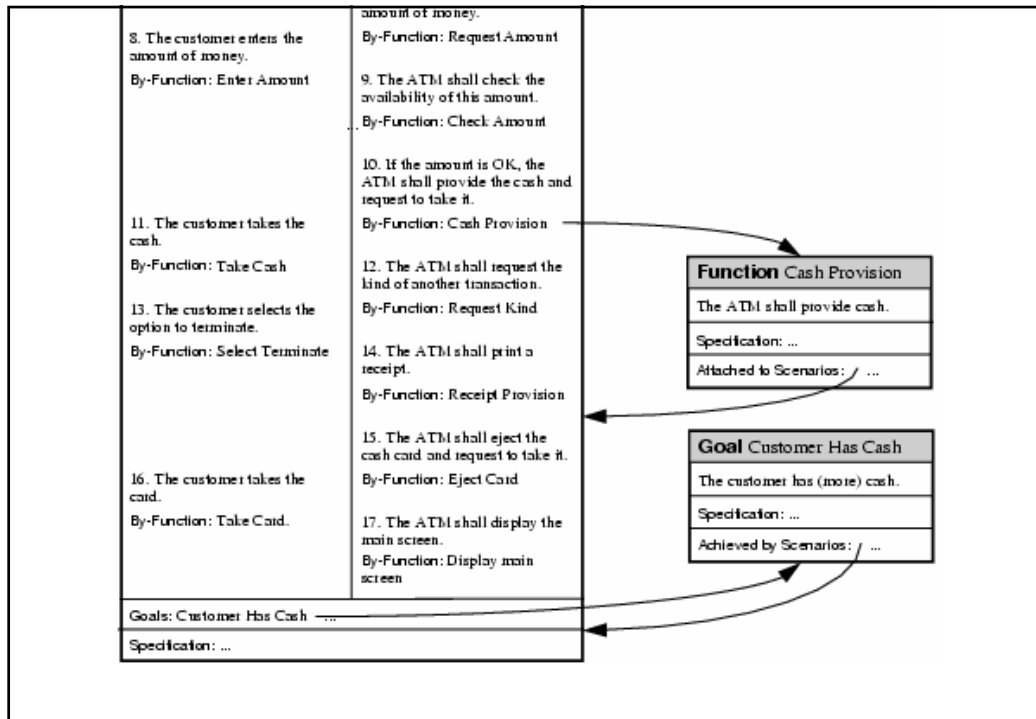
Member Has Video for Rent *By-Scenario* *Rent Available Video*

Member Has Video for Rent:

A member of the video store has a copy of a video title from the store for rent.



Institute of Computer Technology



Use cases – Video Store example

■ Use Case **Rent Video**:

1. Brief Description

This use case allows a member of the video store to rent one or more videos. This rental process is supported both by the VSS software system running on a PC and a clerk employed by the video store.

2. Flow of Events

2.1. Basic Flow of Events

<< See scenario **Rent Available Video** in natural language and/or UML sequence diagram. >>

2.2. Alternative Flows of Events

2.2.1. Identification not successful

3a1. If the identification is not successful, VSS shall ask again for identification.

3a2. The member tries to identify himself/herself to VSS again.

Use cases – Video Store example (cont.)

2.2.2. No available title wanted

- 4a1. If the member does not want to rent any of the available titles, he/she initiates the termination of the transaction.
- 4a2. VSS shall issue a good-bye message and terminate the transaction.

2.2.3. ...

3. Special Requirements

3.1. Performance Requirements

3.1.1. Response time

VSS shall respond to any user input within a maximum of 10 seconds 90% of the time.

...



Institute of Computer Technology

Use cases – Video Store example (cont.)

4. Preconditions

4.1. Running software

The VSS software is up and running on the PC.

4.2. Clerk available

A clerk is on duty at his/her desk, where the PC is located.

5. Postconditions

5.1. Video rent

The member has a video copy of a wanted title for rent.

5.2. Rental booked

The rental is booked in VSS.

5.3. Numbers updated

The number of available video copies of the given title is updated in VSS.



Institute of Computer Technology

Use cases – Video Store example (cont.)

6. Extension Points

6.1. Identification success

Step 3 in scenario *Rent Available Video*

6.2. Title selection

Step 4 in scenario *Rent Available Video*

...

7. Relationships

This use case communicates with the actors Member and Clerk.

8. Use-Case Diagrams

<< See next slide. >>

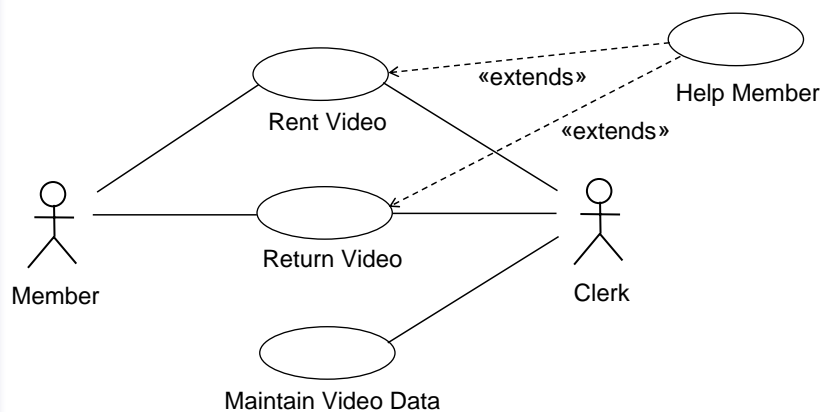
9. Other Diagrams

None

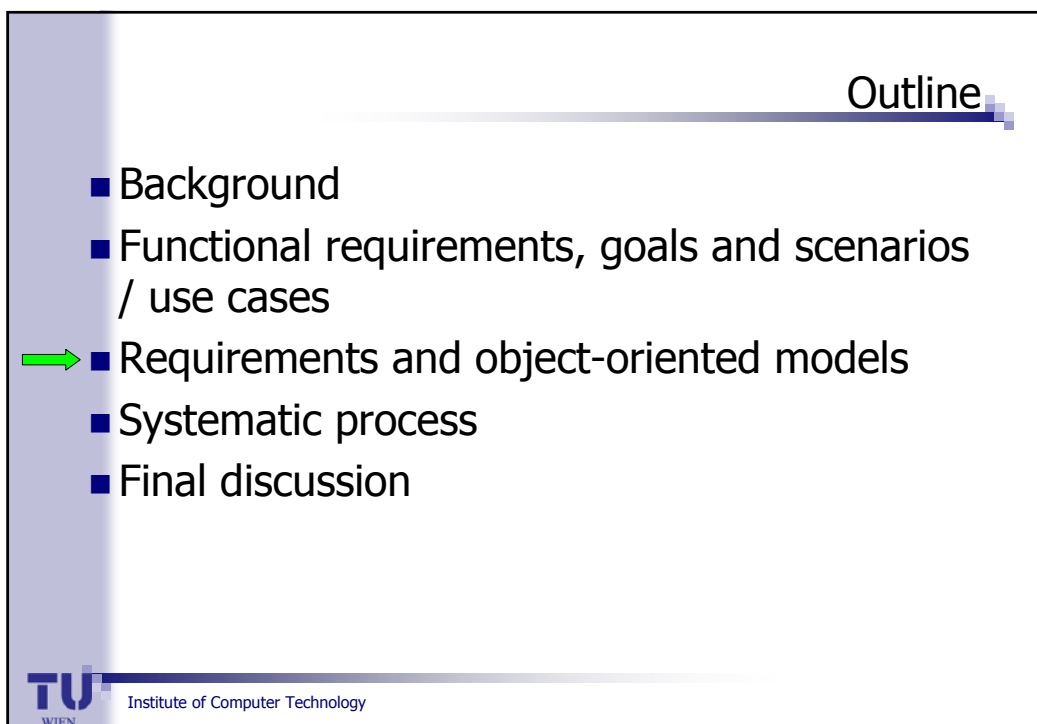
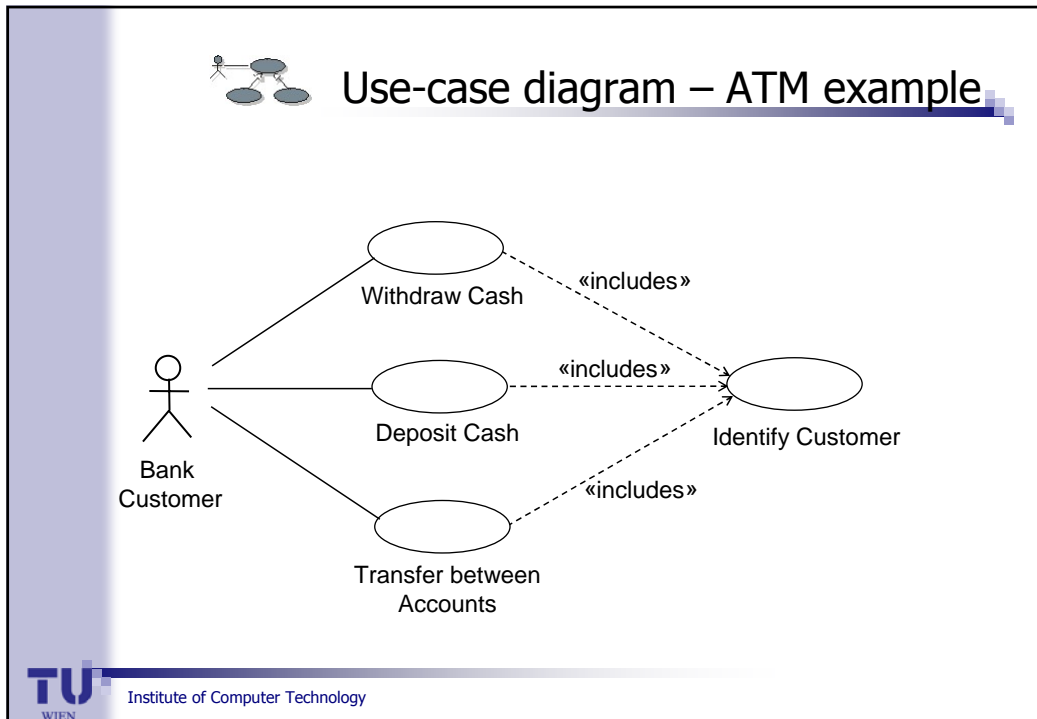


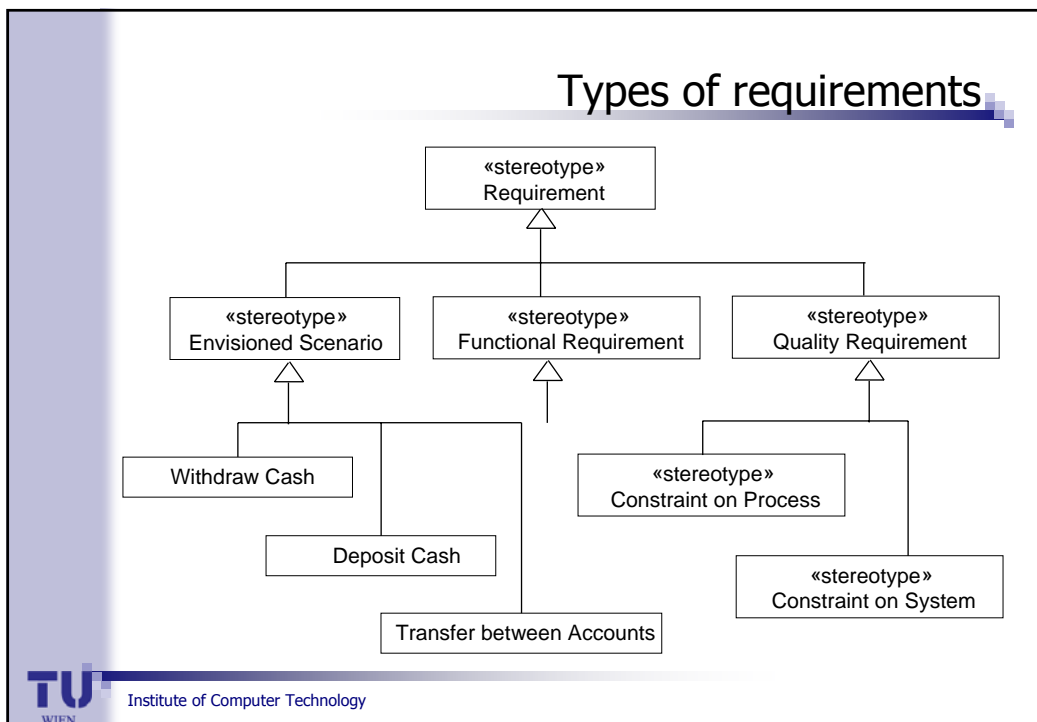
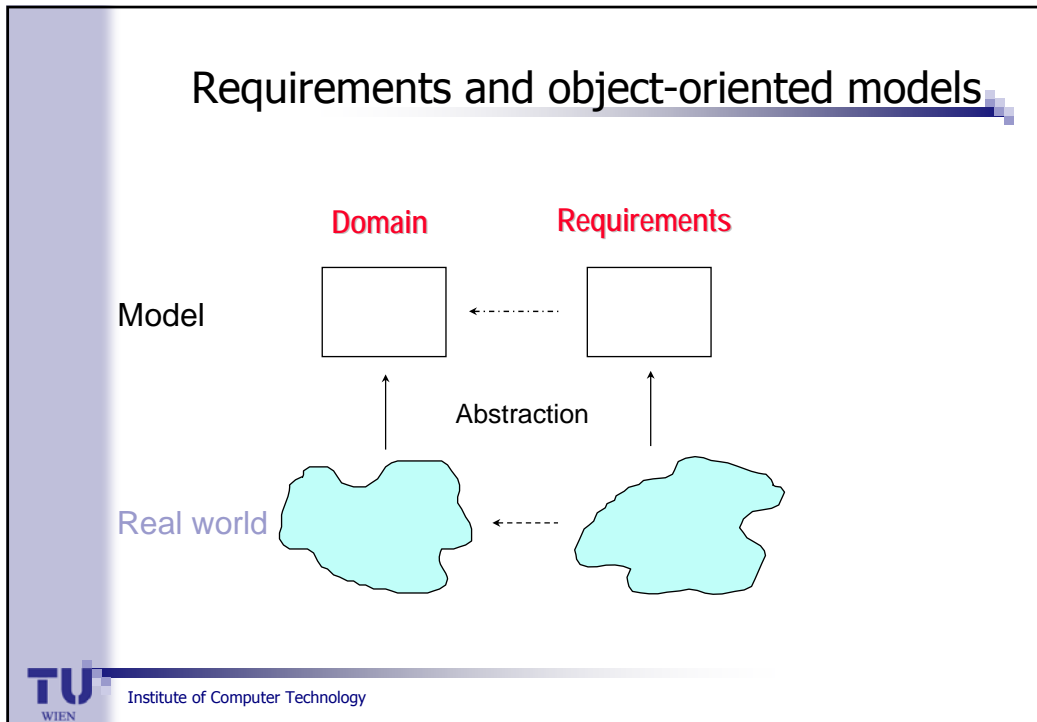
Institute of Computer Technology

Use-case diagram – Video Store example



Institute of Computer Technology





Types of requ. – Constraints on process

- Specific development process to follow
- Specific programming language for implementation
- Specific tools to be used
- Specific hardware to be used
- Political issues
- Time to market
- Terms of delivery
- Cost



Institute of Computer Technology

Types of requ. – Constraints on system

- Performance
- Reliability
- Security
- Safety
- Portability
- Maintainability
- Reusability
- Interface
- Usability



Institute of Computer Technology

Conflicts between Quality Requirements

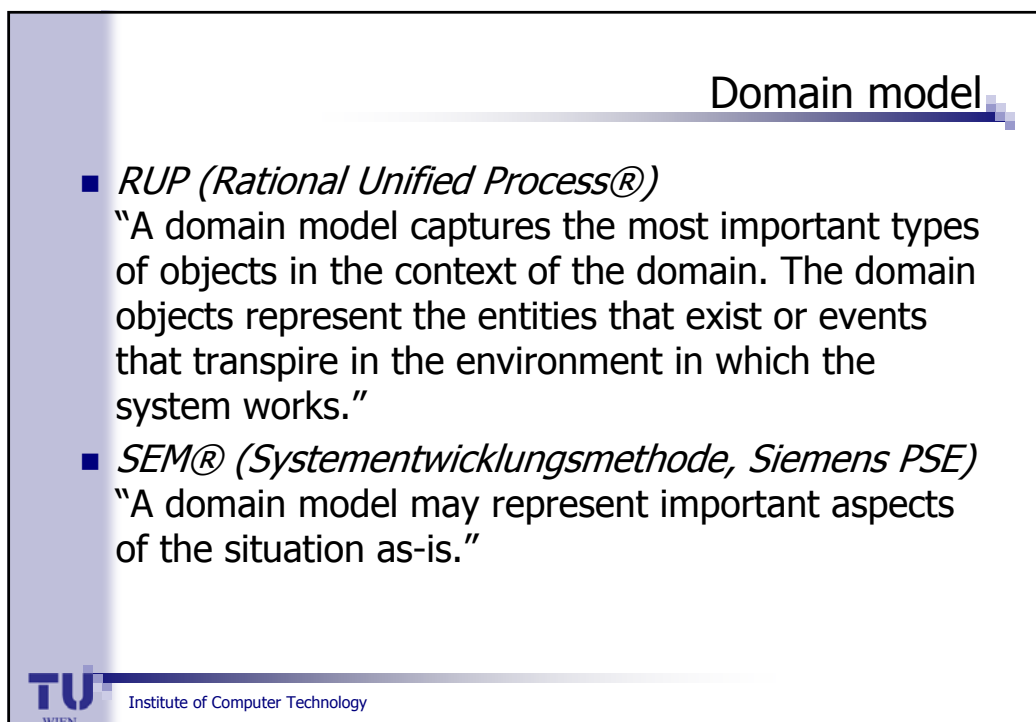
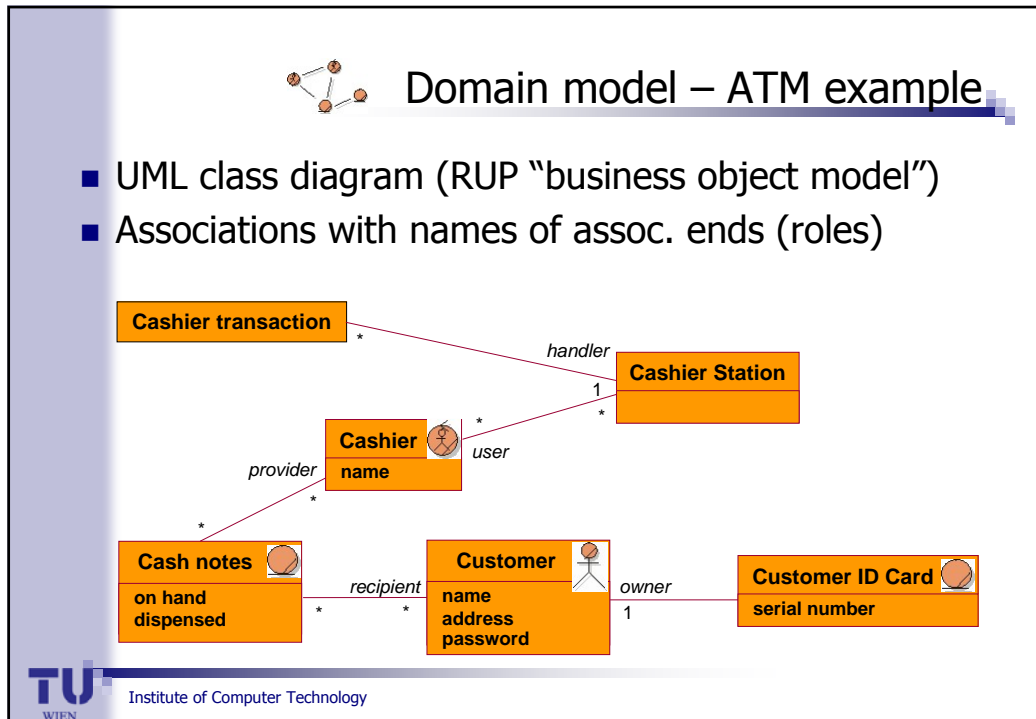
- VSS example
 - VSS shall allow direct access to member data.
 - VSS shall protect member data from illegal access.
- Usability vs. Security
- Trade-off
- Common in complex systems



Types of requ. – Which “system”?

- Requirements for proposed **system to be built**
Software (and hardware) system
- Requirements for **composite system**
Including human users and business artifacts



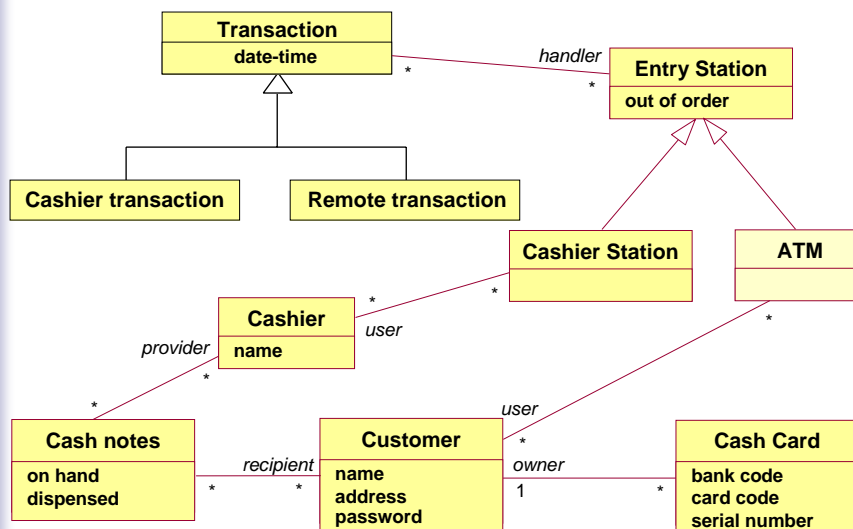


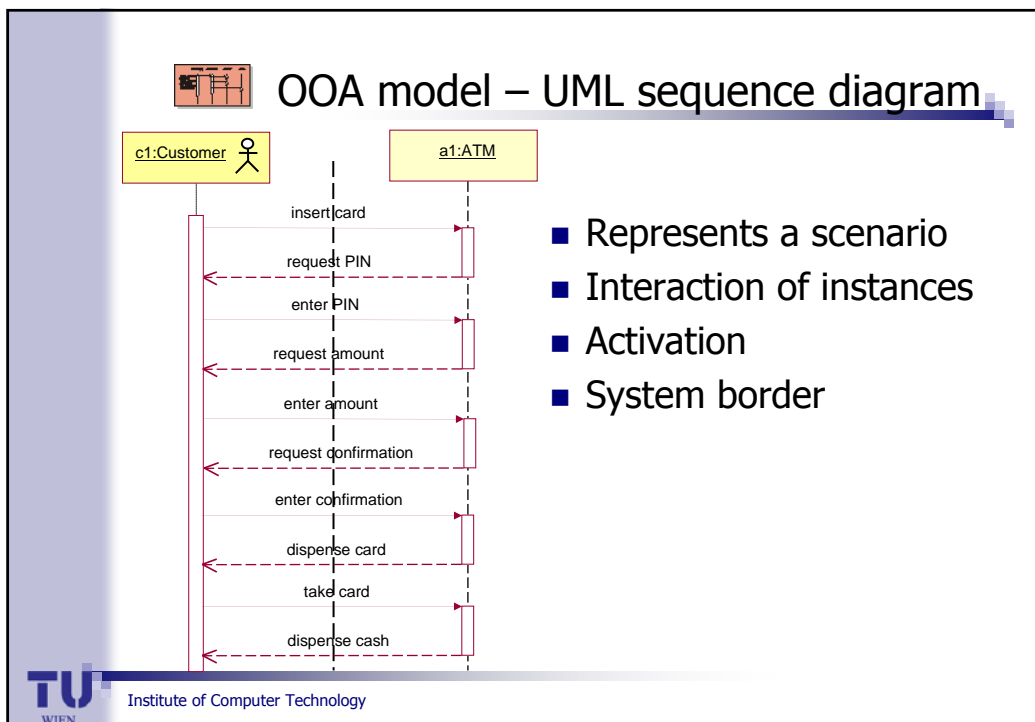
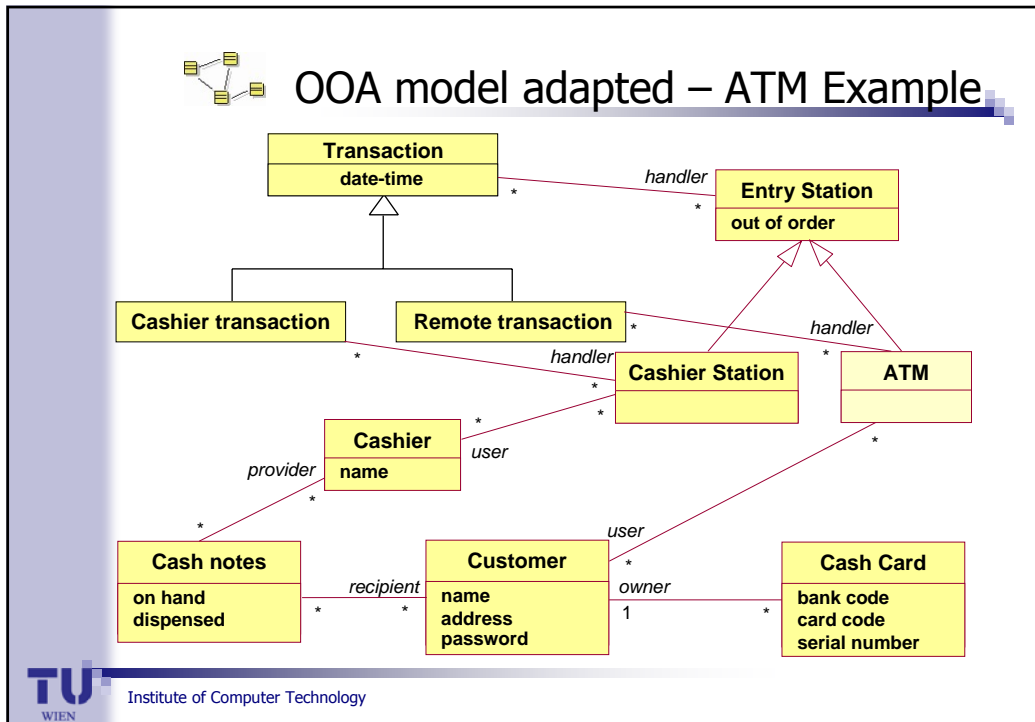
What is analysis?

- *Larman, 2002*
 “Analysis emphasizes an *investigation* of the problem and requirements, rather than a solution.”
- Object-oriented analysis is analysis in terms of (analysis) object classes. ☰



OOA model – ATM Example







OOA model – RUP

- **Analysis Model:**
“An object model describing the realization of use cases, and which serves as an abstraction of the [Artifact: Design Model](#). The Analysis Model contains the results of use case analysis, instances of the [Artifact: Analysis Class](#).”
- “Analysis classes represent an early conceptual model for ‘things in the system which have responsibilities and behavior’.”



OOA model – UP Larman

- “The **Analysis Model** is perhaps not ideally named, as it is actually a kind of design model. In conventional usage, ..., an analysis model suggested essentially a domain object model—an investigation and description of domain concepts. But the UP “Analysis Model” is an early version of the UP Design Model—it describes collaborating software objects with responsibilities.”





OOA model – SEM

■ Purpose

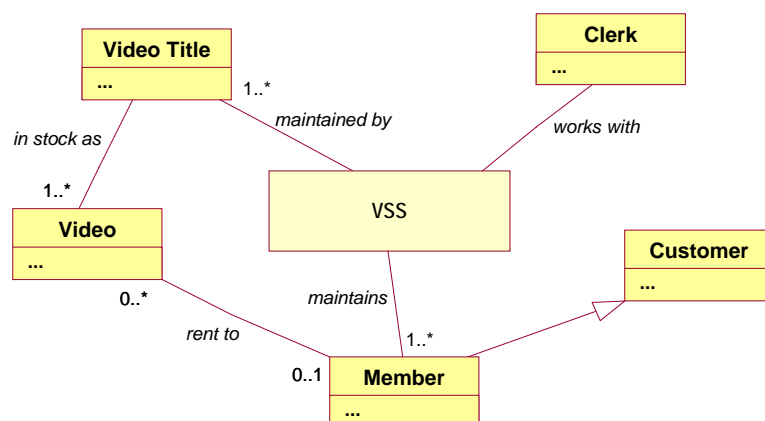
“An OOA model facilitates a better understanding of the application domain as it shall be after the implementation of the product. In this sense, it shall contribute to the specification of the problem and thus of the requirements.”

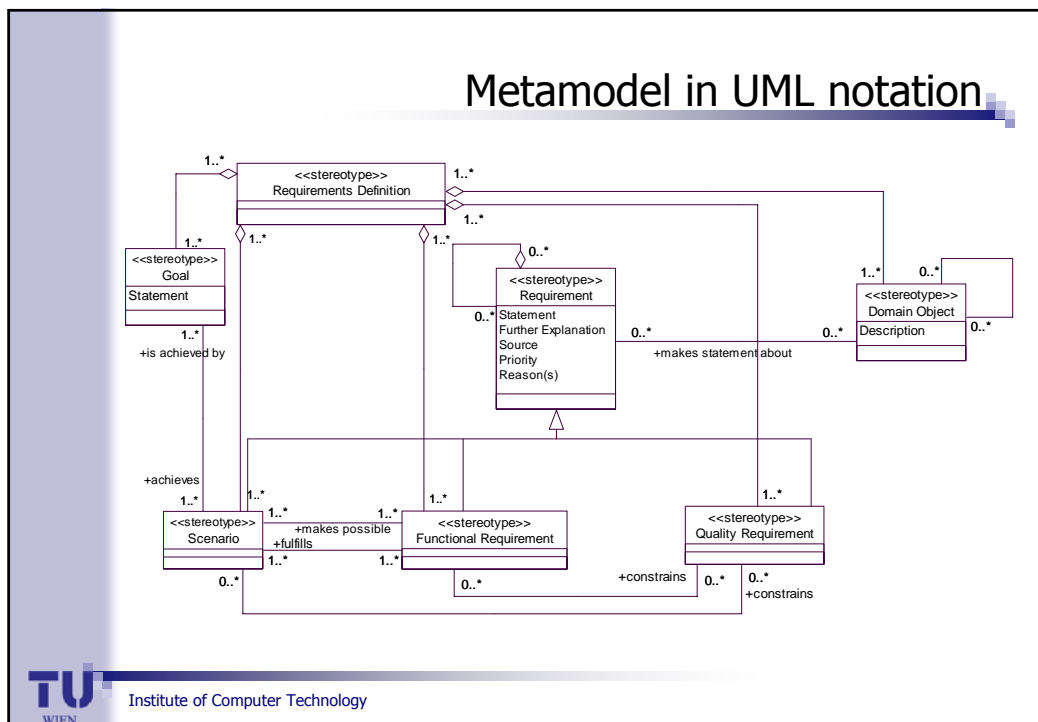
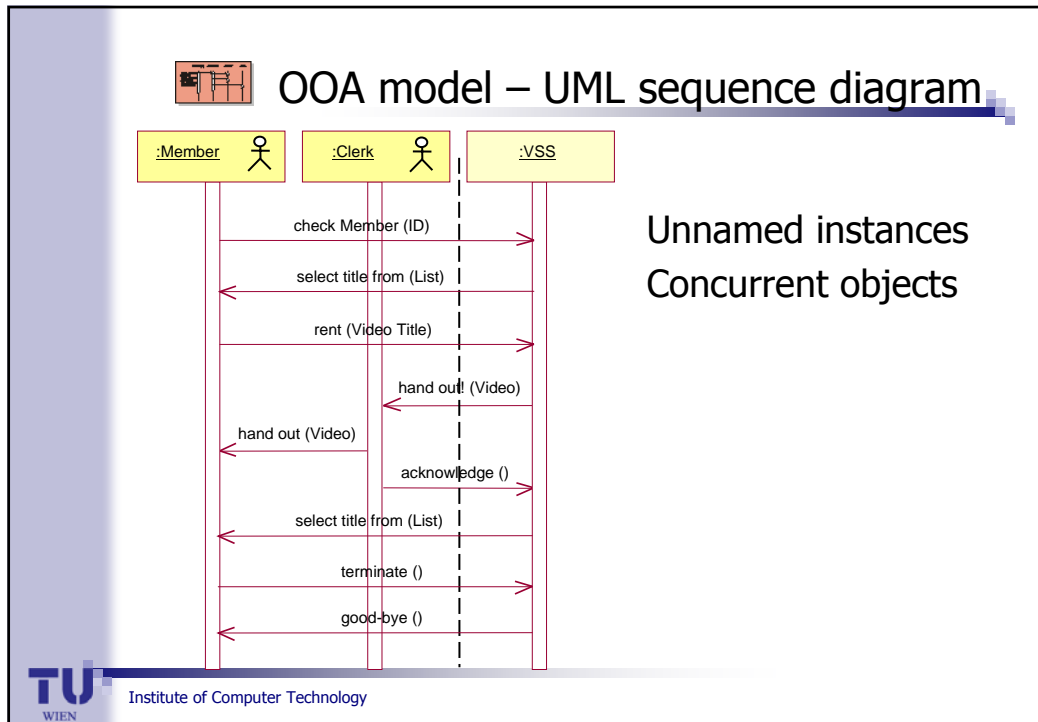
■ Content

“In an OOA model the to-be situation of the real world is modeled, where the product to be built will be integrated.”



OOA model – Video Store example





Outline

- Background
- Functional requirements, goals and scenarios / use cases
- Requirements and object-oriented models
- ■ Systematic process
- Final discussion

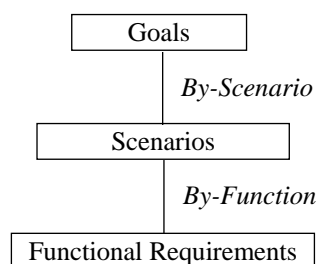


Institute of Computer Technology

Systematic process

Idea: navigation in the metamodel graph

Excerpt:

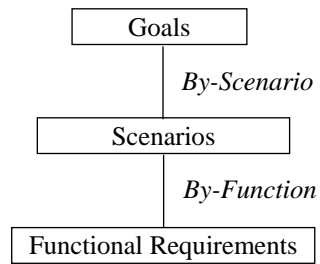


Institute of Computer Technology

Systematic process

Idea: navigation in the metamodel graph

Excerpt:

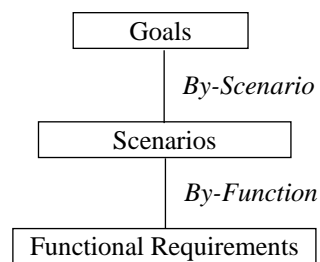


Institute of Computer Technology

Systematic process

Idea: navigation in the metamodel graph

Excerpt:



What is known already?

Old system or system to be built?



Institute of Computer Technology

Systematic process – Given goals

1. If some goal is known from the old system, then figure out whether this is still a goal in the new system that will include the system to be built.
E.g., Meeting a Friendly Person, Customer Has Cash.
2. If some goal is known for the new system, then try to link it to one or more scenarios for the new system that are already known.
E.g., Customer Has Receipt – Get Cash from ATM.
3. If some goal that is known for the new system cannot be linked to any scenario for the new system, then develop one or more such scenarios and link them to the goal.
E.g., Customer Has Cash – Get Cash from ATM.



Institute of Computer Technology

Systematic process – Given scenarios

1. If some scenario is known from the old system, then determine the goals that are achieved through it.
E.g., Get Cash from Human Cashier – Customer Has Cash.
2. If some scenario is known from the old system, then try to develop an analogous scenario for the new system.
E.g., Get Cash from Human Cashier – Get Cash from ATM.
3. If some scenario is known for the new system, then try to link it to one or more goals and, each action contained in it to one or more functions for the new system that are already known.
E.g., Get Cash from ATM – Customer Has Cash – Cash Provision.



Institute of Computer Technology

Systematic process – Given scenarios (cont.)

4. If some scenario that is known for the new system cannot be linked to any goal for the new system, then determine one or more goals and link them to the scenario.
E.g., Get Cash from ATM – Customer Has Cash.
5. If one or more actions contained in some scenario that is known for the new system cannot be linked to any function for the new system, then develop one or more such functions and link them to the actions of this scenario.
E.g., Get Cash from ATM – Receipt Provision.



Systematic process – Given functional requ.

1. If some function is known from the old system, then figure out whether this is still a required function in the new system that will include the system to be built.
E.g., finger prints – Cash Card Acceptance, Cash Provision.
2. If some function is known for the new system, then try to link it to one or more actions contained in scenarios for the new system that are already known.
E.g., Check Amount – Get Cash from ATM.
3. If some function that is known for the new system cannot be linked to any action contained in any scenario for the new system, then develop one or more such scenarios and link one or more actions contained in them to the function.
E.g., money transfer between accounts.



Systematic process (cont.)

- Partial sequences of steps selected according to what is known – agenda
- Both model-driven and data-driven
- Successful termination – agenda empty
- Improvement of
 - Completeness
 - Non-redundancy
 - Understandability
- But no guarantee



Institute of Computer Technology

Outline

- Background
- Functional requirements, goals and scenarios / use cases
- Requirements and object-oriented models
- Systematic process
- ■ Final discussion



Institute of Computer Technology

Summary and Conclusion

- Objects, goals, scenarios and functional requirements can be combined.
- OO domain modeling is useful.
- OO modeling can support requirements engineering and, therefore, systems engineering.



Institute of Computer Technology

Thank you for your attention!

???



Institute of Computer Technology

Selected work of this tutorial presenter

- Kaindl, H., A Practical Approach to Combining Requirements Definition and Object-Oriented Analysis, *Annals of Software Engineering*, 3, 1997, pp. 319–343.
- Kaindl, H., Difficulties in the Transition from OO Analysis to Design, *IEEE Software*, Sept./Oct. 1999, pp. 94–102.
- Kaindl, H., A Design Process Based on a Model Combining Scenarios with Goals and Functions, *IEEE Transactions on Systems, Man, and Cybernetics (SMC) Part A* 30(5), 2000, pp. 537–551.
- Kaindl, H., Is Object-Oriented Requirements Engineering of Interest?, *Requirements Engineering*, 10, 2005, pp. 81–84.
- Kaindl, H., A Scenario-Based Approach for Requirements Engineering: Experience in a Telecommunication Software Development Project, *Systems Engineering*, 8, 2005, pp. 197–210.

