
LUT
University

Software Engineering Models and Modeling

Recap of Software Engineering Fundamentals + Intro to Software Modeling

Antti Knutas

Materials courtesy of Sommerville; part of Software Engineering teaching package; used with permission

What is software engineering?

Software engineering is an engineering discipline that is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use.

Using appropriate theories and methods to solve problems bearing in mind organizational and financial constraints.

All aspects of software production. Not just technical, but also project management, tool development, method development, and modelling to support software production.

Fundamental principles

(adapted from Sommerville)

- Systems should be developed using managed and understood development processes.
- Dependability and performance are important for all types of system.
- Understanding and managing software specification and requirements (what the software should do) are important.
- Where appropriate, reuse software that has been developed rather than write new software.
- Software engineers have responsibilities to the profession, society, and the stakeholders or communities they interact with.

Why is it important?

More and more individuals and society rely on advanced software systems. As professionals, we need to be able to produce reliable, trustworthy, and acceptable systems economically and quickly.

It is usually cheaper, in the long run, to use software engineering methods and techniques for software systems rather than just write personal programming projects.

Software modelling allows understanding all parts of software production, software systems, and systematically plan how they relate to each other.

Brief Introduction to Software Modelling

Introduction to software models

- Model is an abstract view of a system (that ignores irrelevant system details)
- Models should condense a mass of information, such as system structure, into a single more understandable view.
- Models should describe reality and assist understanding it, not act as a limitation.
- Complementary system models can demonstrate context, interaction, structure, and behaviour.

We'll briefly touch on formal models later in the course, but they are not the focus.

System modeling

System modeling is the process of developing abstract models of a system, with each model presenting a different view or perspective of that system.

System modeling has now come to mean representing a system using some kind of graphical notation, which is now almost always based on notations in the Unified Modeling Language (UML).

System modelling helps the analyst to understand the functionality of the system and models are used to communicate with customers.

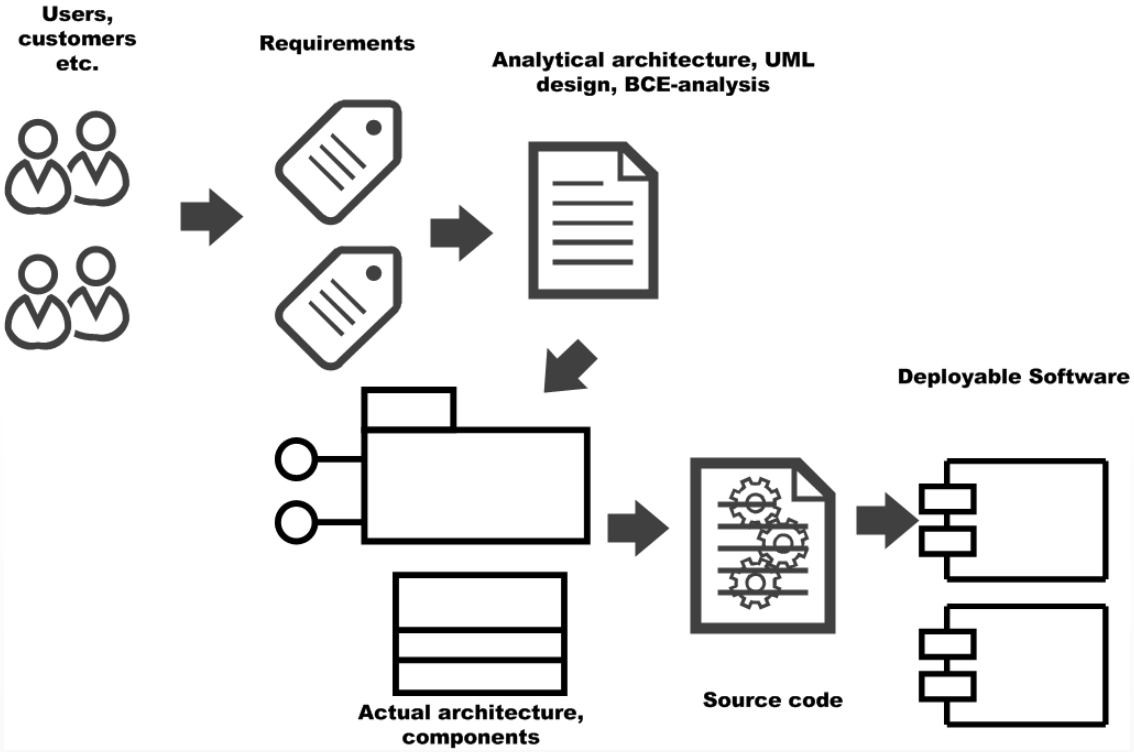
Key points

A model is an abstract view of a system that ignores system details. Complementary system models can be developed to show the system's context, interactions, structure and behavior.

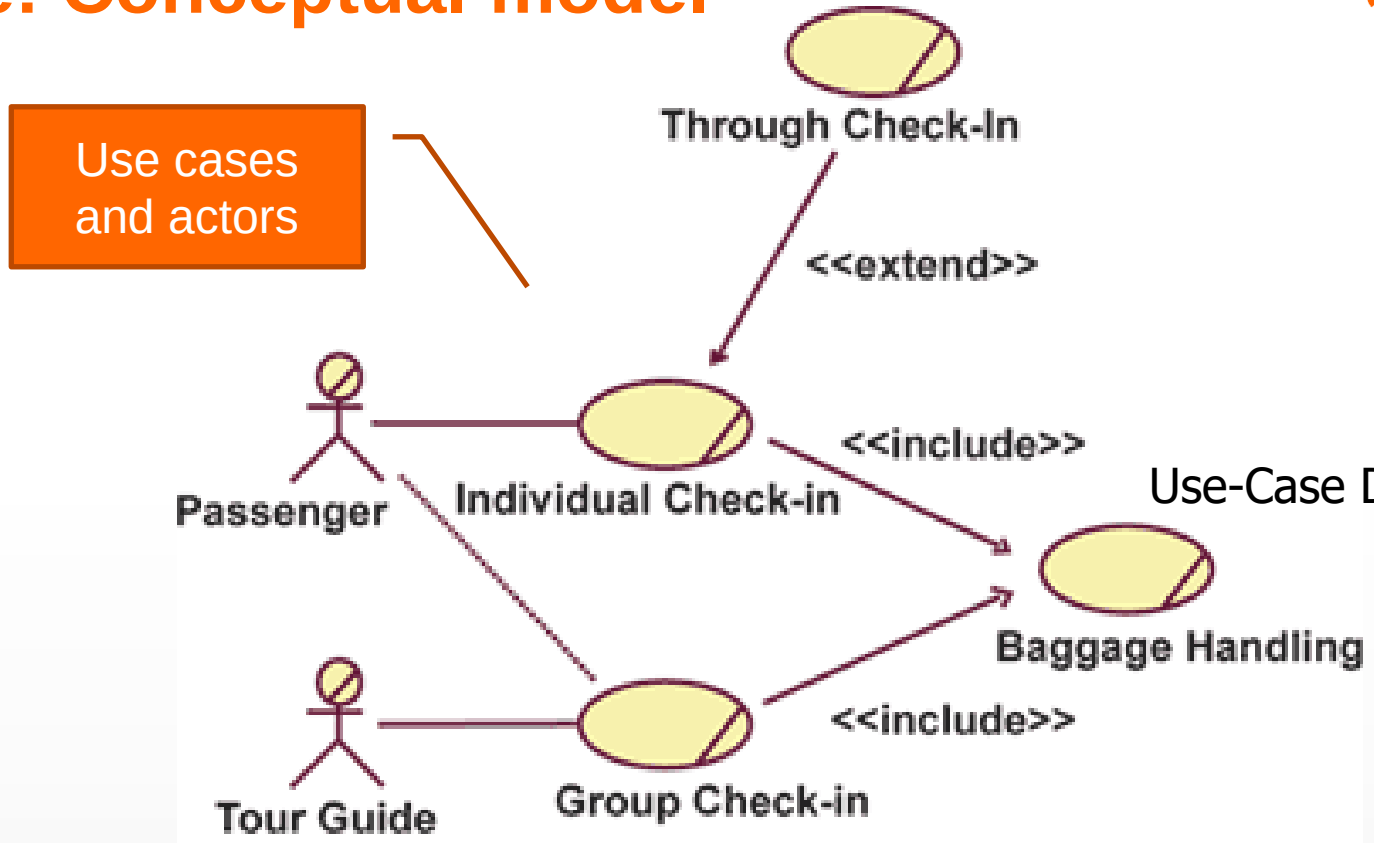
There are diverse types, such as:

- Context models show how a system that is being modeled is positioned in an environment with other systems and processes.
- Use case diagrams and sequence diagrams are used to describe the interactions between users and systems in the system being designed. Use cases describe interactions between a system and external actors; sequence diagrams add more information to these by showing interactions between system objects.
- Structural models show the organization and architecture of a system. Class diagrams are used to define the static structure of classes in a system and their associations.

One SWE process that involves modelling

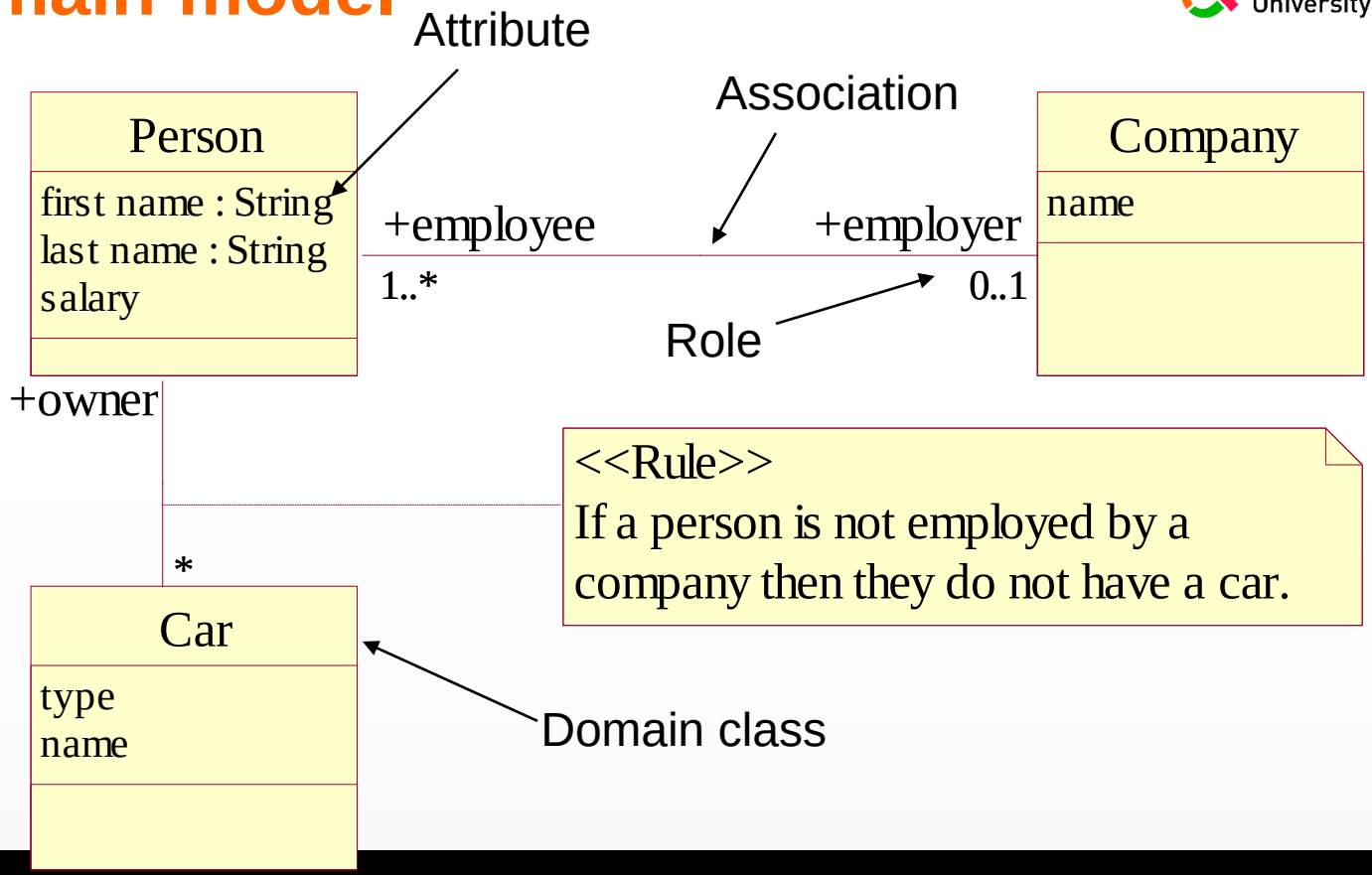


Example: Conceptual model

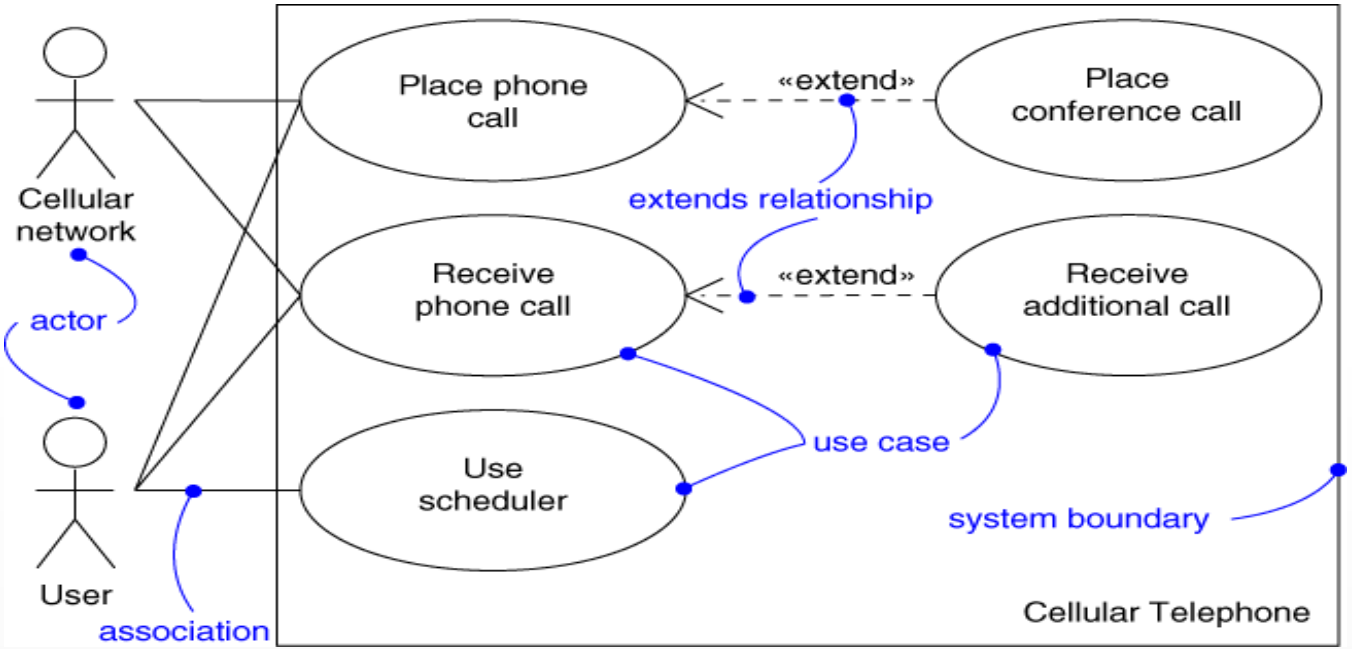


Use-Case Diagram

Example: Domain model

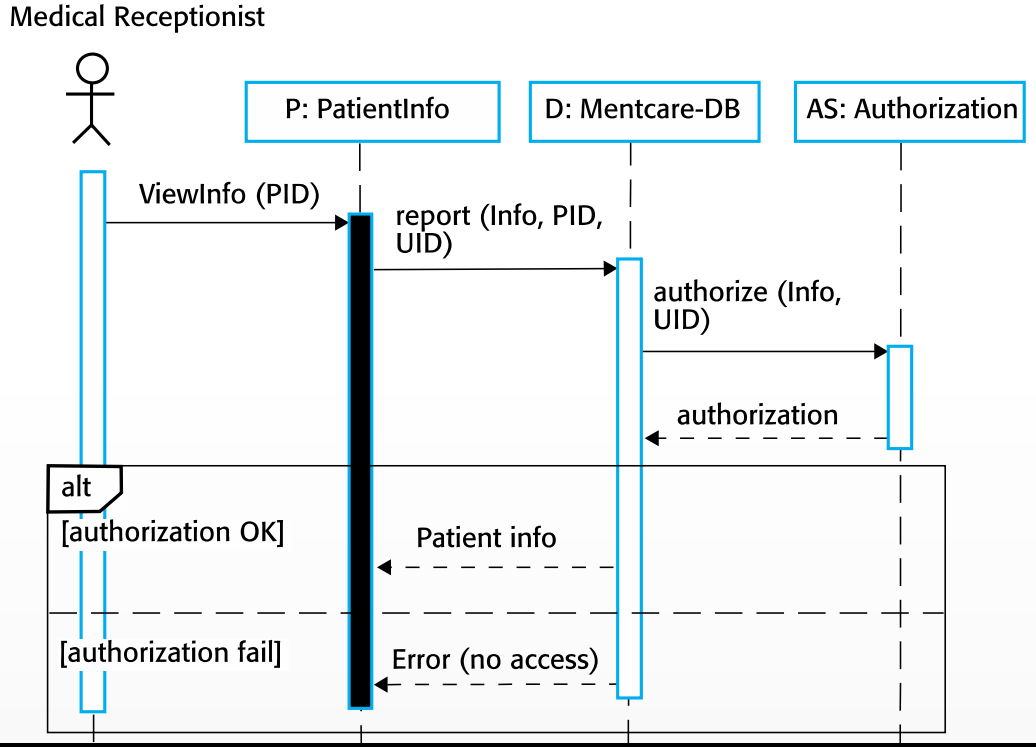


Example: Use case diagram

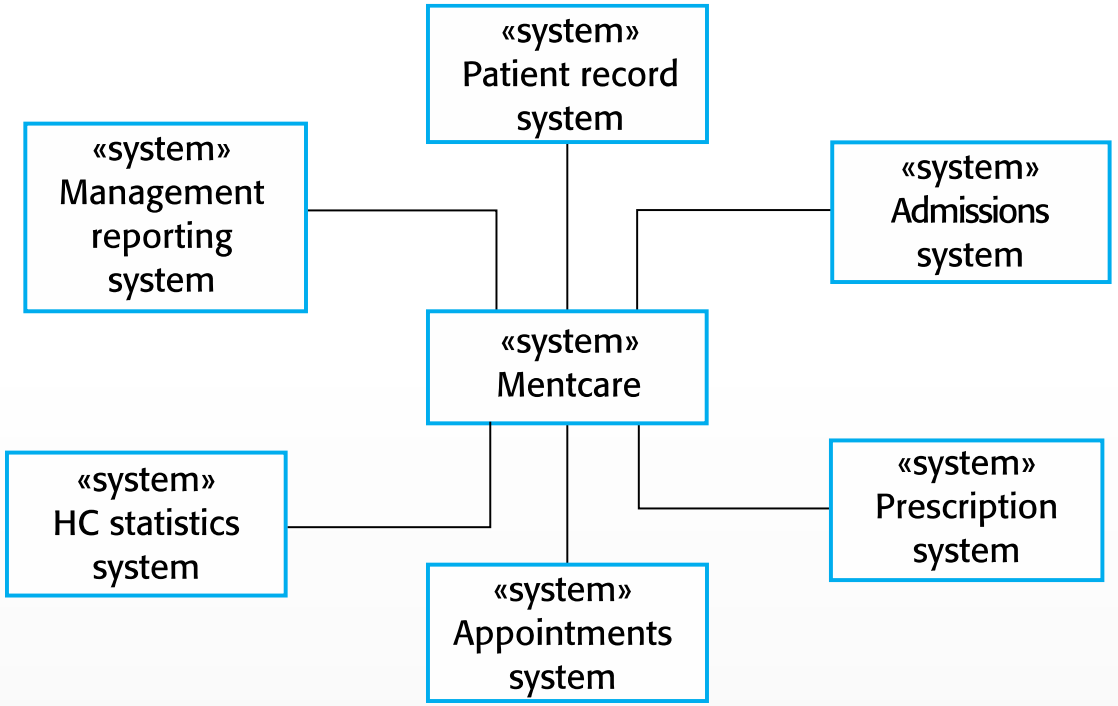


Example: Sequence diagram

Modelling interactions between actors and objects.



Example: Context model



Next week

- Let’s discuss process models
- Introducing the first weekly assignment
- First exercises and (voluntary, optional) team matchmaking
  - Matchmaking => come to the class and ask if someone would like to work with you
  - All exercises have same content. Attend the one that suits you most (Lappeenranta, Lahti, or online)