

Abstraktion und Modellierung

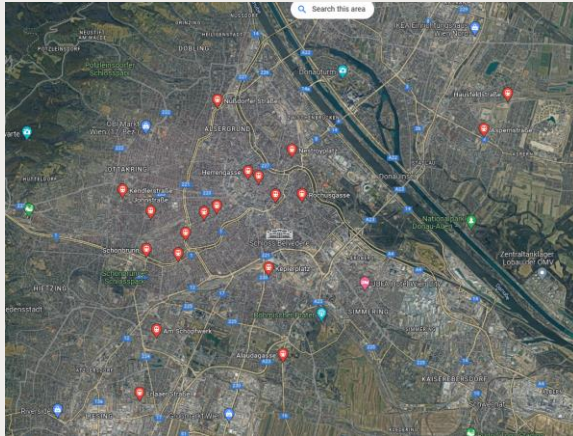
Computational Thinking

Abstraktion

Abstraktion Definition

A way of expressing an idea in a specific context while at the same time suppressing details irrelevant in that context.

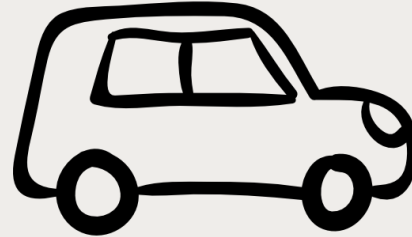
== Weglassen von Details, die in dem momentanen Kontext unwichtig sind.



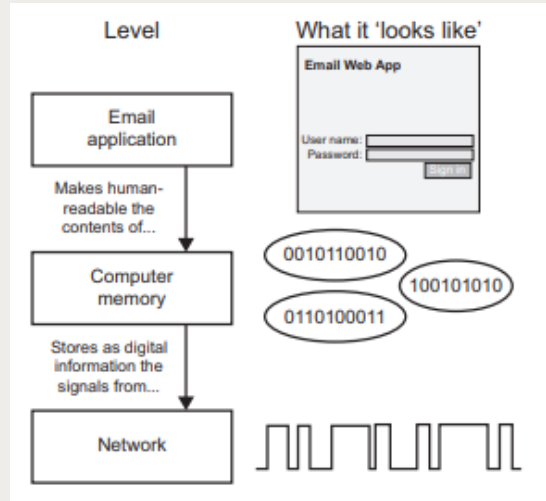
Abstraktion: Auto

A way of expressing an idea in a specific context while at the same time suppressing details irrelevant in that context.

== Weglassen von Details, die in dem momentanen Kontext unwichtig sind.

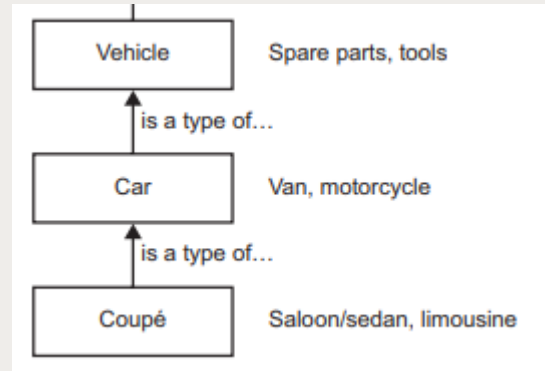


Abstraktionslevel



Das Abstraktionslevel und der Kontext bestimmt welche Details in der Abstraktion wichtig sind.

Abstraktionslevel



Das Abstraktionslevel und der Kontext bestimmt welche Details in der Abstraktion wichtig sind.

Abstraktion beim Programmieren

Was macht der Computer bei $x = 10$?

1. Der Computer sucht sich im RAM freien Speicherplatz
2. Er speichert den Wert 10 dort ab
3. Die Adresse des Speicherplatzes wird für x im Python-Programm verwendet

Abstraktion beim Programmieren

Eine Programmiersprache ist eine Abstraktion

1. Python Code von einem Compiler gelesen
2. Der Compiler transformiert den Code in Byte Code
3. Die Python Virtual Machine (Interpreter) transformiert den Byte Code in maschinenlesbare Instruktionen
4. Diese Instruktionen können von der CPU im Computer gelesen und ausgeführt werden

Abstraktion beim Programmieren

Datentypen

Table 10.1 List of some of the most commonly used types in Python

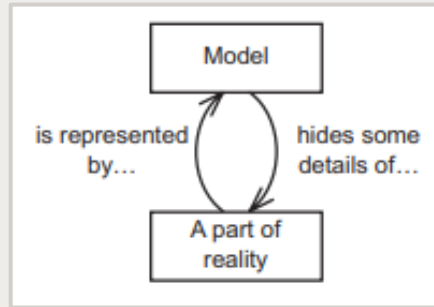
Type	Example	Comments
int	<code>answer = 42</code>	Integer. Stores whole numbers.
float	<code>half = 0.5</code>	Floating point numbers. In other words, it stores real numbers (see the box below).
bool	<code>is_ready = True</code>	Stores Boolean values, i.e. True or False.
list	<code>ages = [32, 31, 27, 31, 39]</code>	Traditionally (though not necessarily) used as a variable-sized sequence of data where all the items have the same type.
tuple	<code>profile = ('Palin', 1943, 'Sheffield')</code>	Traditionally (though not necessarily) used as a fixed-sized sequence of data where the items have mixed types.
set	<code>evens = {2, 4, 6, 8}</code>	An unordered collection of items. No item can appear more than once in a set.
dict	<code>num_chapters = {'bible': 929, 'hobbit': 19}</code>	Dictionary. Stores a collection of items (similarly to a list), but gives each item a unique, meaningful identifier to facilitate later retrieval.
str	<code>name = 'Spock'</code>	String. Holds a piece of text.

Modellierung

Was sind Modelle?

== eine Art von Abstraktion.

Repräsentiert ein System, ein Verhalten, die Realität,



Warum brauchen wir Modelle?

- Besseres Verständnis durch Vereinfachung
- Bessere Kommunikation
 - Technik <--> Projektmanagement <--> Kunden
- Bessere Dokumentation

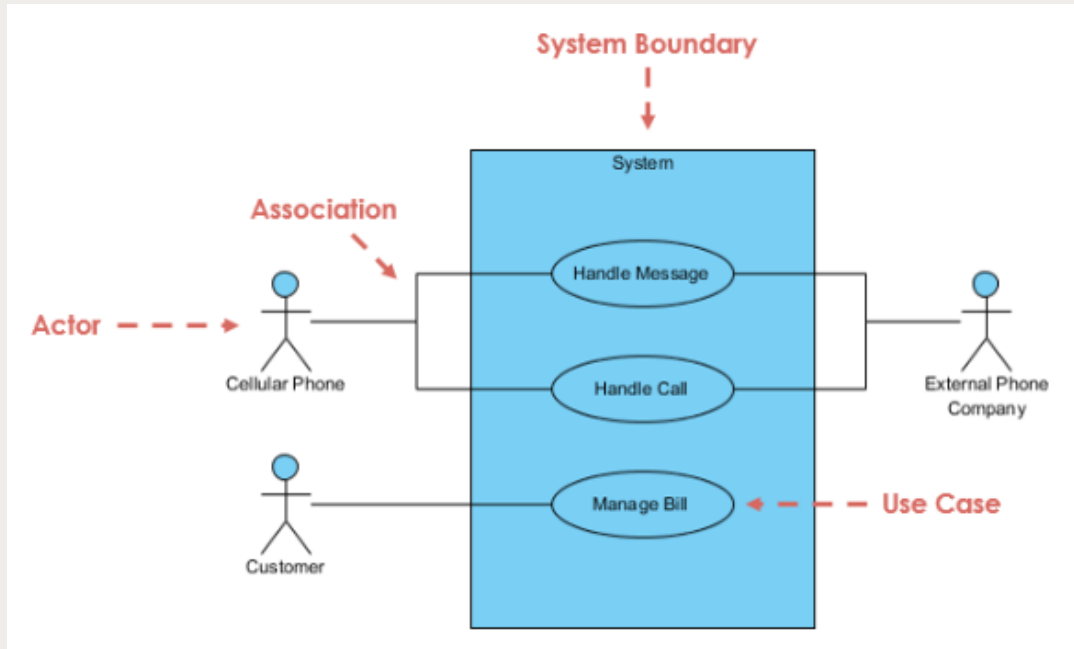
Modellierung im Programmieren mit UML

- UML (Unified Modelling Language) = Grafische Modellierungssprache zur Spezifikation, Konstruktion, Dokumentation und Visualisierung von Softwaresystemen
- Modelle von UML
 - Class Diagram
 - **Use Case Diagram**
 - **Activity Diagram**
 - Behaviour Diagram
 - Timing Diagram
 - Component Diagram
 - ...

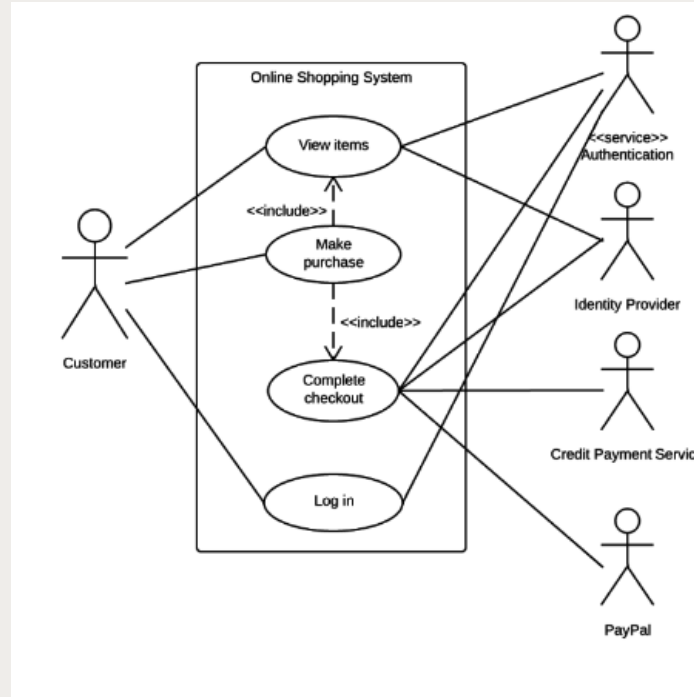
Use Case Diagramm

- Ein Use Case Diagramm beschreibt wie verschiedene User mit dem System interagieren können
(Inputs und Outputs des Systems zu Usern)
- Wird benutzt für:
 - Herausfinden, wie das System Usern helfen kann
 - Beschreibt wie das System mit Organisationen, Usern und externen Systemen interagiert
 - Repräsentiert typische Use Cases


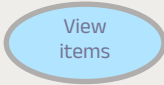
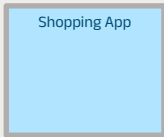
Use Case Diagramm

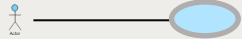
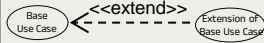
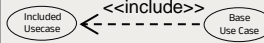



Use Case Diagramm Beispiel



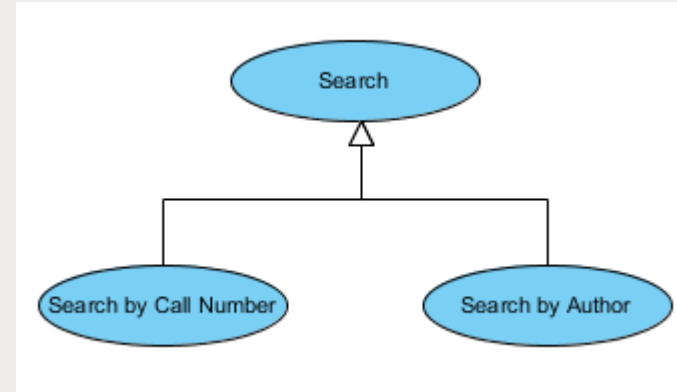
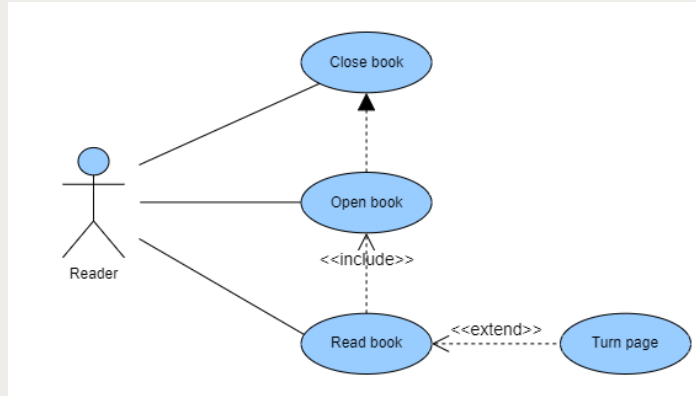
Use Case Diagramm - Aufbau

	Aktor Interagiert mit dem System.
	Use Case Bildet Prozess/Funktion/Use Case im System ab.
	Systemgrenzen Bildet die Grenzen des Systems ab. Prozesse liegen innerhalb.

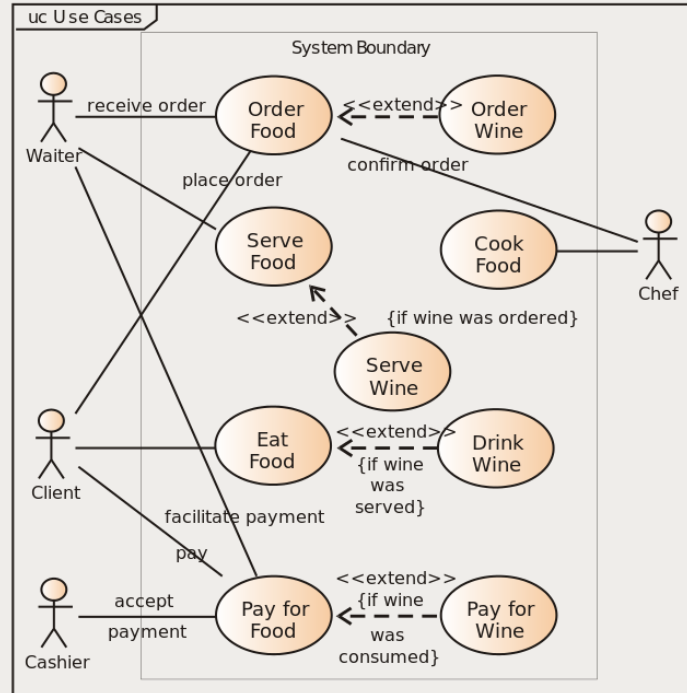
	Verbindung Interaktion zwischen Actor und System.
	<<extend>> Der Use Case ist optional und kommt nach dem Basis-UseCase.
	<<include>> Der Use Case ist verpflichtend und Teil vom Basis-UseCase.
	Parent-Child Beziehung Parent ist eine Abstraktion der Children. Children sind Spezialisierungen des Parents.

Use Case Diagramm

(include, extend, parent-child)

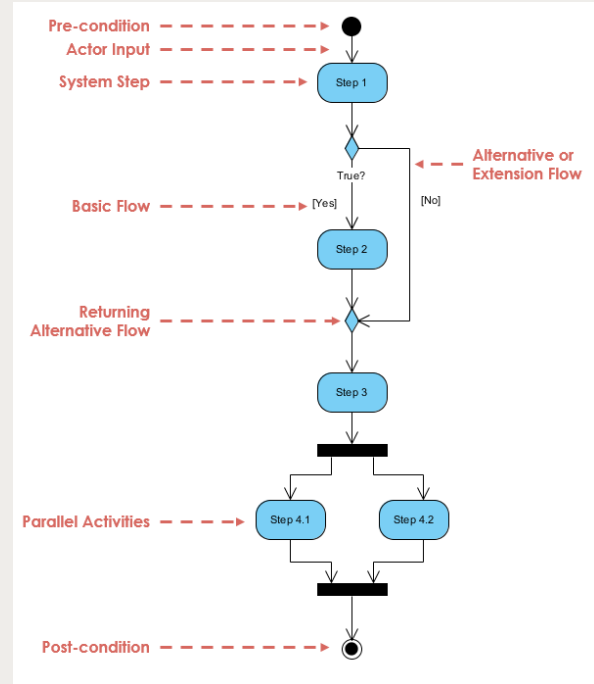


Use Case Diagramm Beispiel

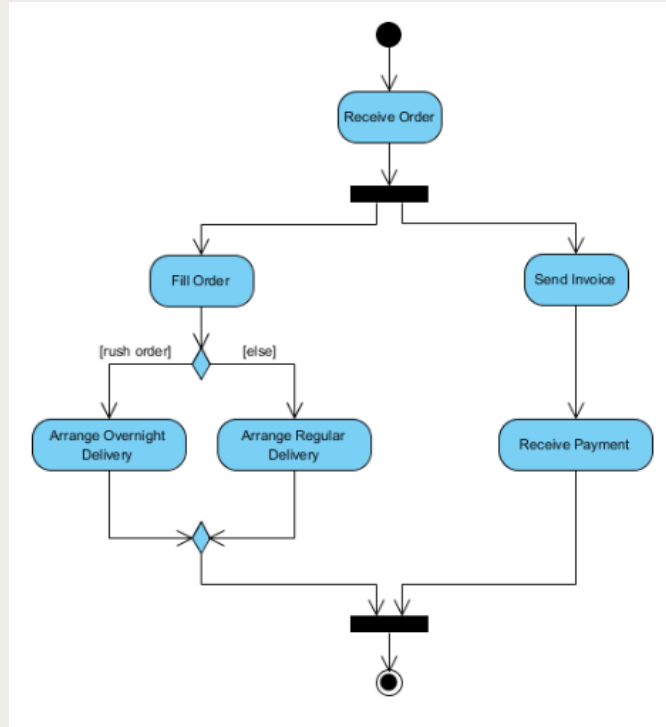


Activity Diagram

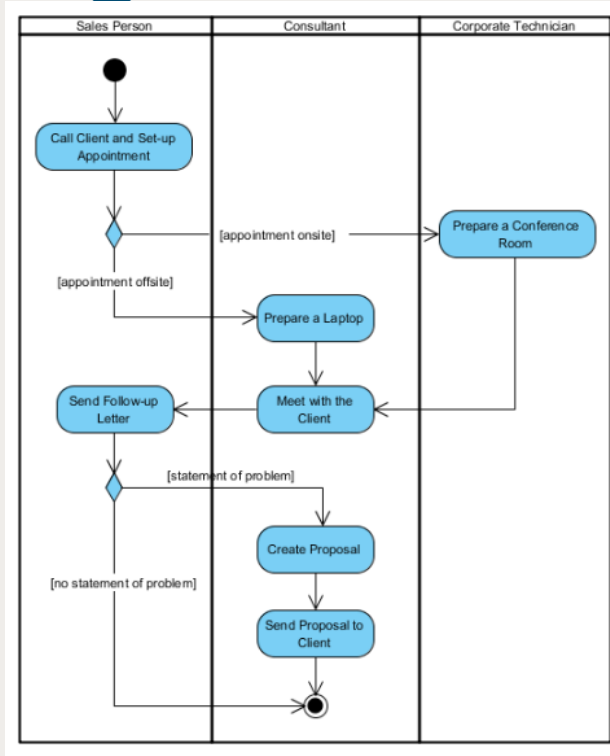
- Anwendungsfälle
 - Stellt Prozesse grafisch da
 - Businessprozess
 - Softwareprozess
 - Business Requirements
- Mit und ohne Swimlanes
 - Swimlanes gruppieren Aktivitäten nach Akteur oder System



Activity Diagram



Activity Diagram – mit Swimlanes



Allgemeine Tipps zu Modellen

- **Abstract:** a good model hides unimportant and irrelevant details, allowing us to concentrate on the important concepts.
- **Understandable:** a good model presents information intuitively and requires less effort to understand than the equivalent code does.
- **Accurate:** a good model must be a true-to-life representation of the system it models.
- **Predictive:** a good model allows us to correctly predict the non-obvious properties of a system.
- **Inexpensive:** it should be cheaper to produce a model than to construct the actual system itself.