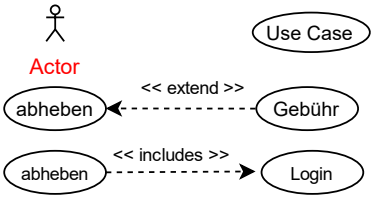
SOFTWARE ENGINEERING / UML - DIAGRAMS / SYMBOLS

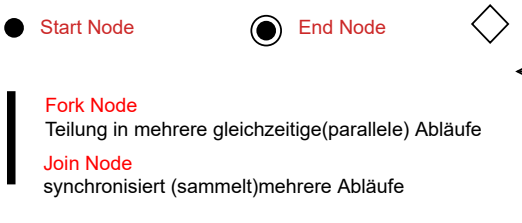
USE CASE DIAGRAM | Behaviour Diagramm



extend = wird verwendet, um optionale oder zusätzliche Funktionen darzustellen, die unter bestimmten Bedingungen zum Tragen kommen, aber **nicht zwingend notwendig** sind

inkludes = dient dazu, **große Use Cases zu vereinfachen**, indem wiederkehrende oder gemeinsame Verhaltensweisen in weitere Use Cases ausgelagert werden.Um doppelte Logik zu vermeiden.
("Login" wird für die use cases 1. Geld abheben und 2. Geld einzahlen und 3. Geld überweisen benötigt)

ACTIVITY DIAGRAM | Behaviour Diagramm



CLASS DIAGRAM | Strukture Diagram

Zugriffsmodifikatoren:

+ public
- private
protected
~ package
Abstract

Multiplizität:

* = 0,1, or mehr
1 = genau 1
2..4 = zw. 2 u. 4 (inkl.)
3..* = 3 oder mehr

Komposition (besteht - aus - Beziehung)
Eine Universität kann nicht ohne Räume existieren. Wenn Universität gelöscht wird,
gibt es auch keinen Raum mehr.

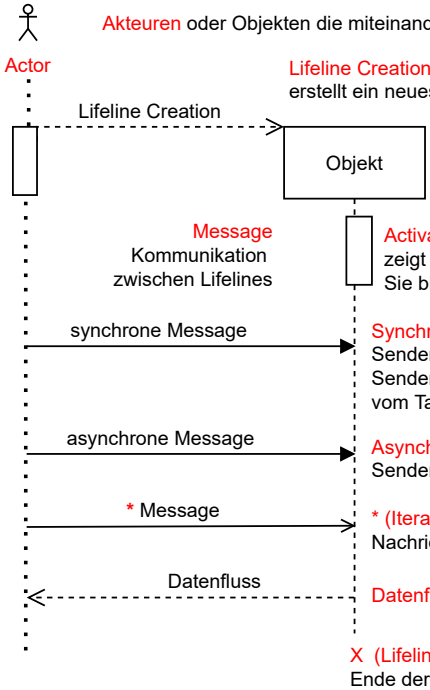
Aggregation(ist Teil von - Beziehung)
Person ist Teilvon Kurs. Aber beide existieren auch unabhängig von einander .

Assoziation (starke Beziehung)
dauerhafte Beziehung zwischen 2 Klassen.
Multiplizität ! Ein Professor kann einen oder mehrere Kurse haben

Abhängigkeit (schwache Beziehung)
Diese Beziehung besteht nur während eine bestimmten Interaktion. Keine Multiplizität !

Generalisierung / Vererbung
Studen und Professor sind Spezialisierungen (Vererbungen)
der allgemeinen Klasse Person

SEQUENZ DIAGRAM | Behaviour Diagramm



SRS / THE SOFTWARE REQUIREMENT SPEZIFICATION
(User and SystemRequirements)

funktional Anforderungen
(werden im **USE-CASE DIAGRAMM** dargestellt)
= beschreiben das Verhalten und die Structur des Systems

nicht funktionale Anfoderungen
Zuverlässigkeit, Reaktionszeit, Speicheranforderungen, Performance, Sicherheit, Skalierbarkeit, Wartbarkeit, etc.
= können oft mehrere funktionale Anforderungen auslösen
= solten in messbaren Größen definiert werden.

RUPP`s METHODE

- **Prozess identifizeren** = nutze Prozesswörter (drucken, speichen, ..) = Das System speichert Daten

- **Aktivität charakterisieren** = unterscheide zwischen
  Systemaktivität: *Das System druckt*
  Benutzerinteraktion: *Das System ermöglicht dem Nutzer, eine Datei zu speichern*
  Schnittstelle: *Das System empfängt Daten von der Bibliothek.*

- **Bestimme Grad der Verbindlichkeit**:
  shall/must: Verpflichtend. = *Das System **soll** die Datei speichern.*
  should: Wünschenswert, aber nicht zwingend
  will/could: Absicht oder zukünftige Möglichkeit.

  Won`t = optional - wird im aktuellen Release nicht inkludiert

→ MOSCOW METHODE