TECHNISCHE UNIVERSITÄT DRESDEN

Requirements Engineering V

Anforderungsmodellierung

Dr. Birgit Demuth

(unter Wiederverwendung der Folien von Ulrike Hammerschall (1))

Themen ReqEng V

- Modellbildung
  - Modelle, Sichten, Notationen
  - Syntax, Semantik und Pragmatik
  - Formalisierung von Sprachen
- Modellierungstechniken
- Entwicklung des Anforderungsmodells
  - Sichten des Anforderungsmodells
  - Notationen und ihre Anwendung
  - Auswahl von Notationen
- Modellgetriebenes Requirements Engineering (MoDRE)

Themen ReqEng V

- Modellbildung
  - Modelle, Sichten, Notationen
  - Syntax, Semantik und Pragmatik
  - Formalisierung von Sprachen
- Modellierungstechniken
- Entwicklung des Anforderungsmodells
  - Sichten des Anforderungsmodells
  - Notationen und ihre Anwendung
  - Auswahl von Notationen
- Modellgetriebenes Requirements Engineering (MoDRE)

Modelle und Sichten

- Ein **Modell** ist eine Abstraktion eines beliebigen Originals. Es beschreibt die für den Einsatzzweck des Modells relevanten Aspekte des Originals.
- **Originale** eines Modells können sein:
  - physische Objekte (z.B. Auto, Buch, Paket),
  - virtuelle Objekte (z.B. Konto, Buchung, Software),
  - Personen und Rollen (z.B. Kunde, Mitarbeiter, Student),
  - Prozesse (z.B. Geld abheben, Projekt durchführen).
- Eine **Sicht** ist ein Modell, welches das Original aus einer bestimmten Perspektive heraus beschreibt.

Abstraktion

- Welche Aspekte im Modell dargestellt werden, hängt von seiner Zielsetzung ab.
- Modelle entstehen durch:
  - Reduktion: Unwichtige Details des Originals werden weggelassen.
  - Abstraktion: Die relevanten Informationen im Original werden verallgemeinert, beispielsweise durch Klassenbildung.

Sprachen und Notationen

- **Sprachen** sind die Grundlage der Kommunikation.
- Ihre schriftliche Repräsentation wird auch als **Notation** bezeichnet.
- Es gibt Sprachen, die
  - nur gesprochen,
  - gesprochen und geschrieben,
  - nur geschrieben werden.
- **Modellierungssprachen** sind reine Schriftsprachen zur Darstellung von Modellen.

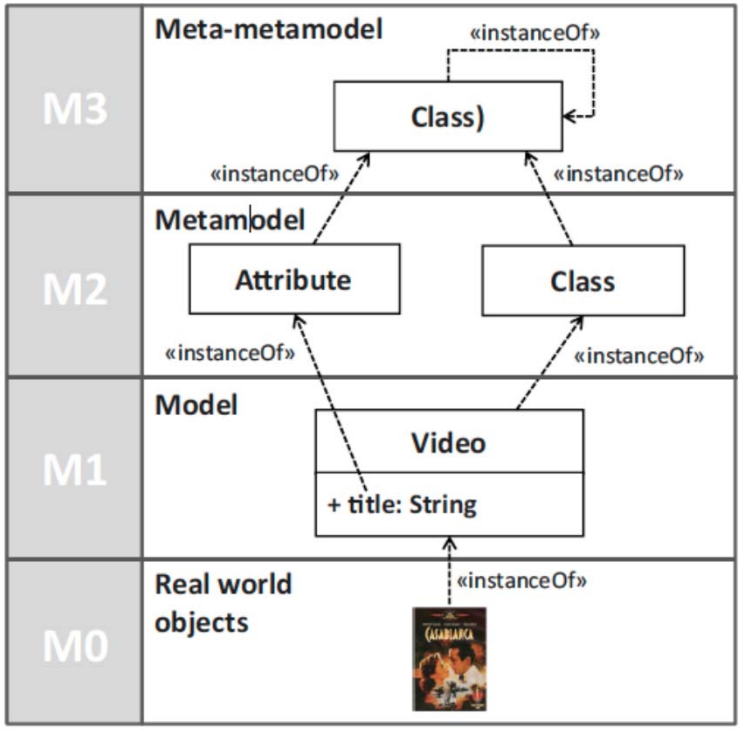
Syntax, Semantik und Pragmatik

- Eine Sprache besteht aus drei Teilen: Syntax, Semantik und Pragmatik.
  - Syntax: Symbole (Alphabet), korrekte Verknüpfung der Symbole (Grammatik).
  - Semantik: Bedeutung der Symbole und ihrer Verknüpfungen.
  - Pragmatik: Üblicher Einsatz der Sprachmittel.

Formalisierung von Sprachen

- Informelle Notationen
  - Syntax und Semantik nicht oder nur informell vorgegeben.
  - Beispiele: freie Diagramme
- Semi-formale Sprachen
  - Syntax vorgegeben, Semantik halbformal beschrieben
  - Beispiele: UML, OCL
- Formale Sprachen:
  - Syntax und Semantik mathematisch formal beschrieben
  - Beispiele: Prädikatenlogik, VDM, Z-Notation

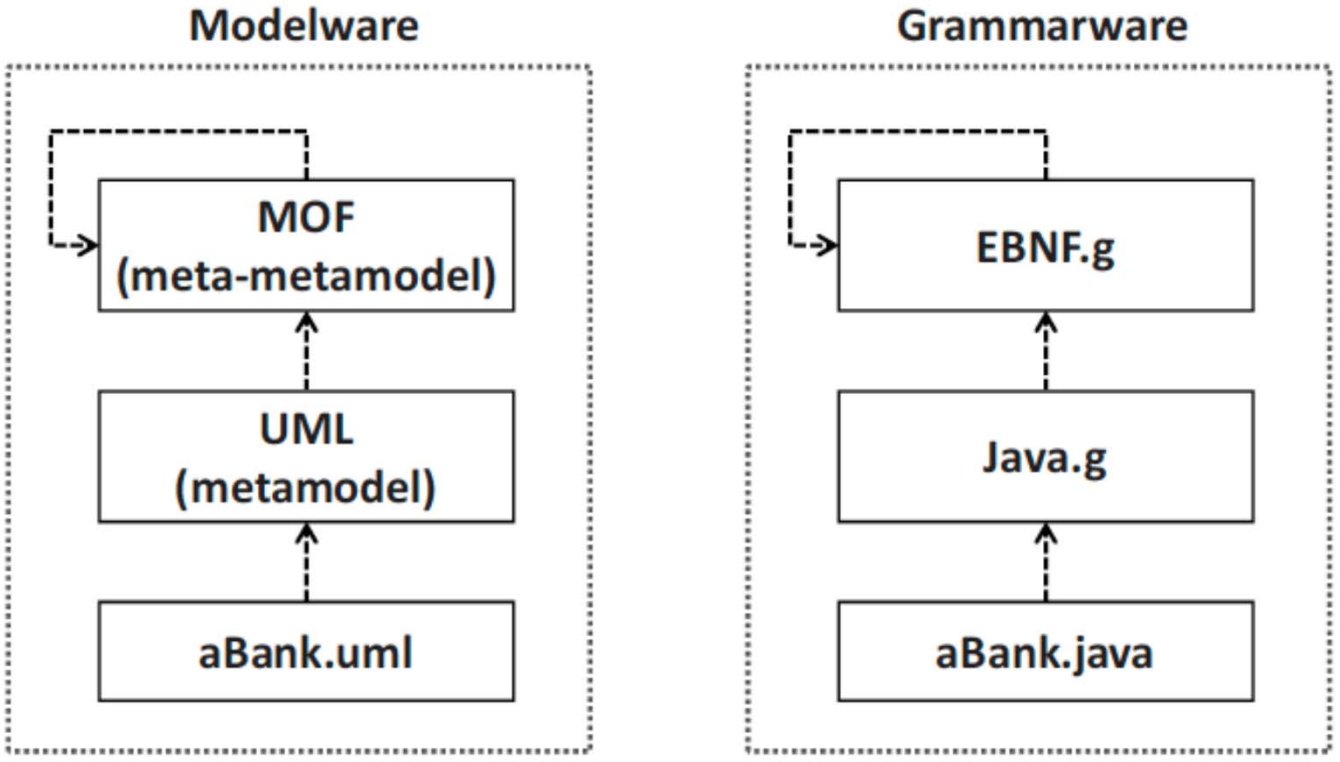
Metamodellierung



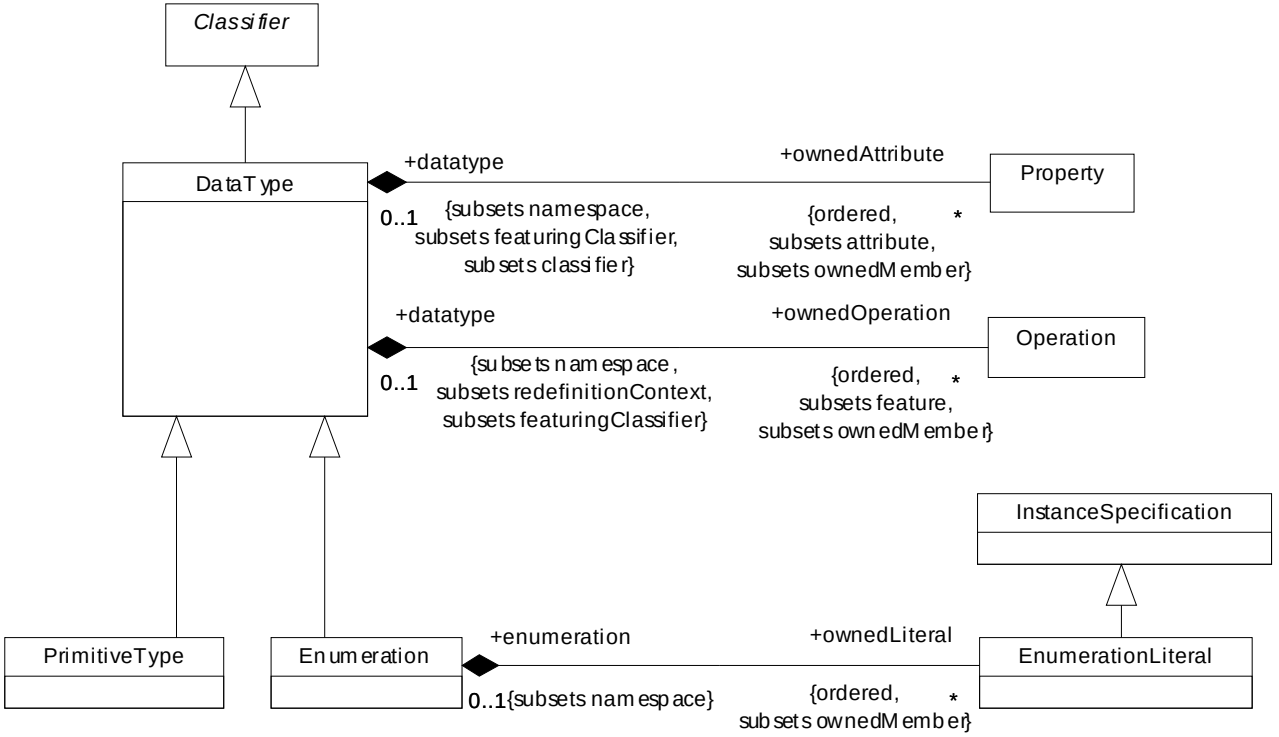
Metamodell

- allgemein Metadaten eines Modells
- d h. Abstraktion von einem Modell
- zur Definition der **Syntax** von Modellierungssprachen
- **Semantik** teilweise durch WFRs (Well-Formed Rules) mit Hilfe der Object Constraint Language (OCL) beschrieben

Technikräume



Beispiel: Metamodell von UML Datentypen



Vorgehen zur Modellbildung

(1) Identifikation und Abgrenzung des zu modellierenden Originals
(2) Festlegung des Einsatzzwecks für das Modell
(3) Auswahl der geeigneten Sichten und Notationen
(4) Festlegung der Abbildungsvorschrift
(5) Modellierung der Sichten

Themen ReqEng V

- Modellbildung
  - Modelle, Sichten, Notationen
  - Syntax, Semantik und Pragmatik
  - Formalisierung von Sprachen
- Modellierungstechniken
- Entwicklung des Anforderungsmodells
  - Sichten des Anforderungsmodells
  - Notationen und ihre Anwendung
  - Auswahl von Notationen
- Modellgetriebenes Requirements Engineering (MoDRE)

Sichten im Anforderungsmodell (bisher)

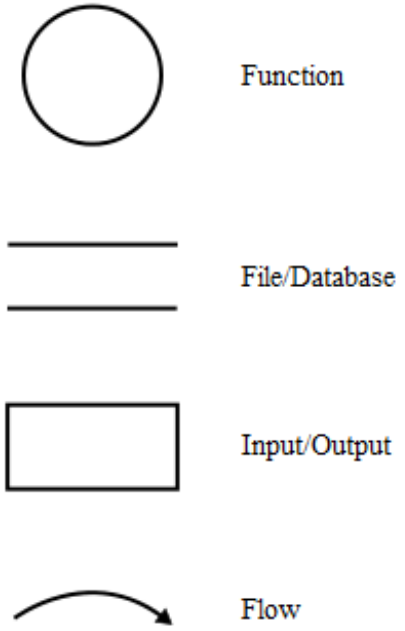
Sicht	Beschreibung	Notationen
Kontextsicht	Modelliert die Einbettung des Systems in seine Umgebung.	Kontextdiagramm
Funktionssicht	Modelliert die funktionale Einbettung des Systems in die existierende Systemlandschaft.	
Struktursicht (Domänenmodell, logische Datenmodelle)	Modelliert strukturelle Zusammenhänge der Anwendungsdomäne. Dies sind fachliche Konzepte mit ihren Eigenschaften und Beziehungen.	
Verhaltenssicht	Modelliert das Verhalten fachlicher Objekte der Anwendungsdomäne.	Entscheidungstabelle
Schnittstellensicht	Modelliert das Schnittstellenverhalten des Systems.	Use Case Diagramm Aktivitätsdiagramm

Historische Sicht auf die Modellierung von Informationssystemen

- Information Algebra (1962) → Information Sets of Items, Relationships zwischen Information Sets, Operationen
- 1970er Jahre: neue Modelle, Standardisierungsbemühungen
  - *What are we modelling?*
  - IFIP TC2 on Software, WG2.6 on Database (frühe 70er Jahre)
  - Standards Planning and Requirements Committee (SPARC) of ANSI/X3: 3-Schichten-Architektur für DB (1975)
  - IFIP TC7 on Information Systems, WG8.1 on formal description and analysis of Information Systems (1977)
  - ACM SIGMOD (1975)/jährliche Konferenzen
  - VLDB series of conferences (1975)
- 1980er Jahre: Suche nach einem gemeinsamen Framework
- 1990er Jahre: Business and Enterprise Modelling-Bewegung
  - *Why are we modelling?*
  - *How are we modelling?*

Auf dem Weg zur Strukturierten Analyse

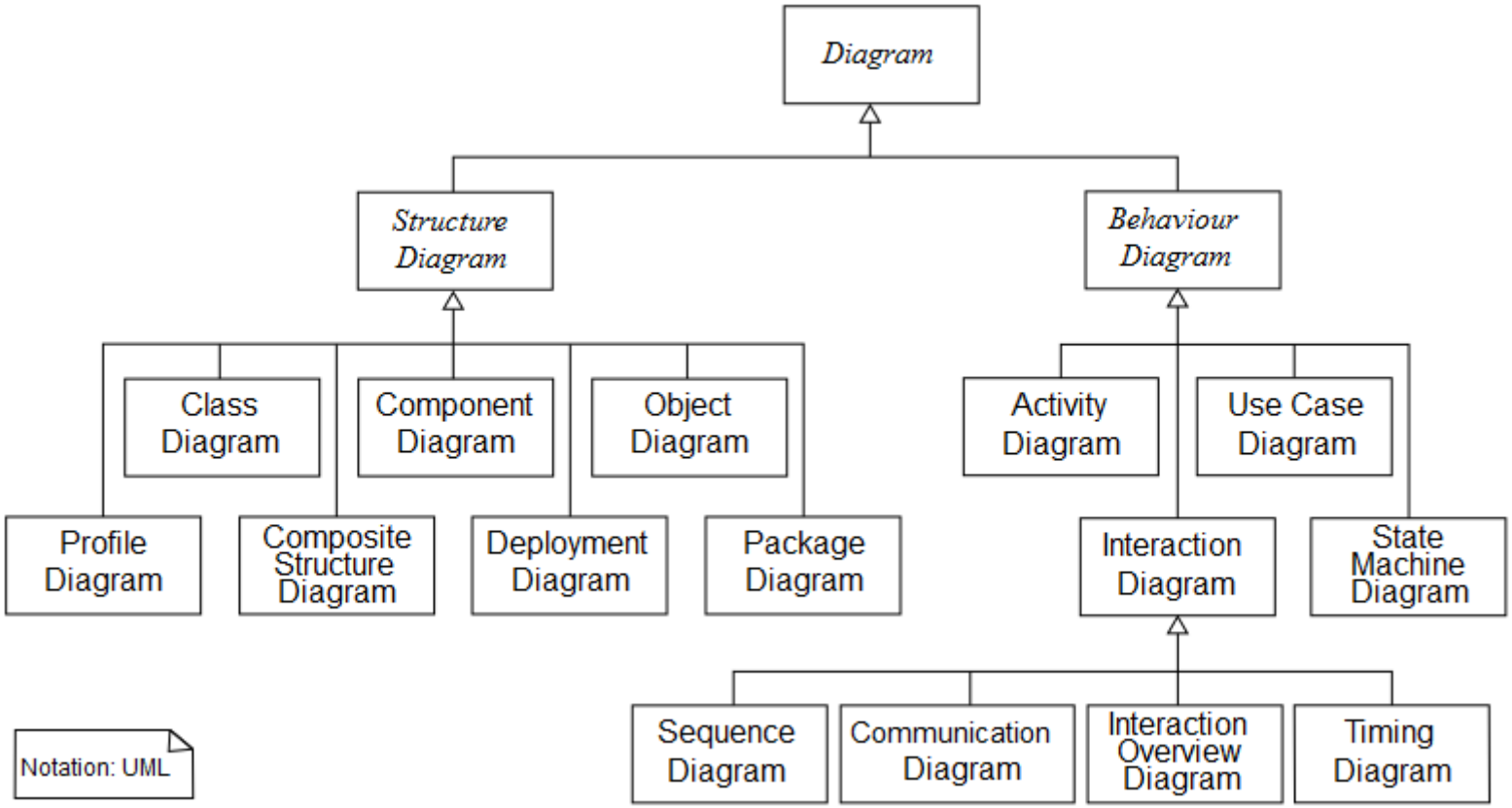
- Structured Programming (Edgar Dijkstra), 1968
- Structured Analysis and Design Techniques (SADT), 1977
  - Funktionen werden schrittweise „top-down“ zerlegt
- Structured Analysis (SA), 1978, Tom de Marco, Ed Yourdon
  - Datenflussdiagramme (DFD)
  - zeigen den Fluss der Daten durch das System und wo sie gespeichert werden



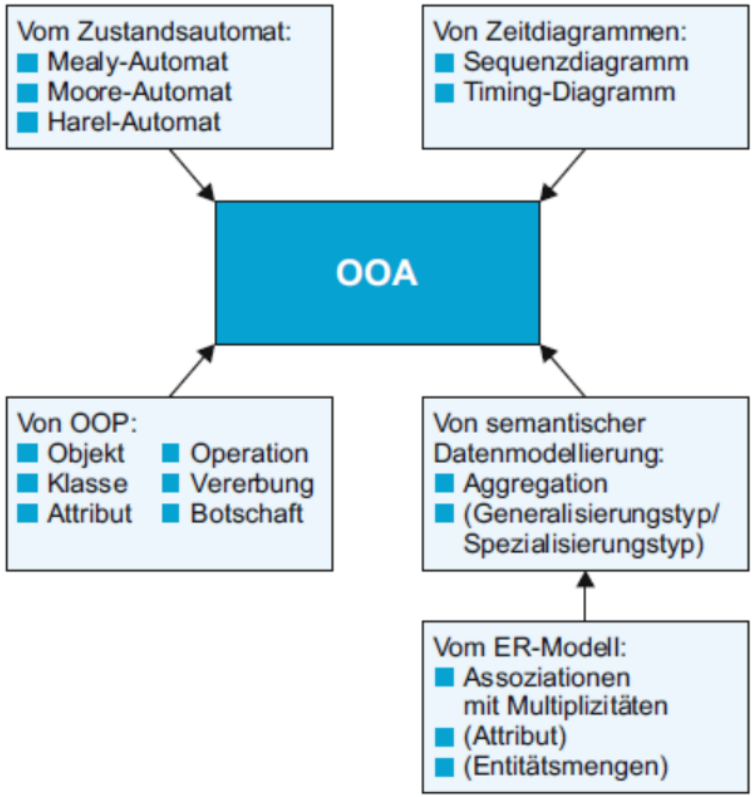
Objektorientierte Modellierungstechniken

- Unified Modeling Language (UML)
  - Grady Booch, Ivar Jacobson, James Rumbaugh @ Rational Software, 1990er Jahre
  - OMG-Standard, 1997
  - derzeitige OMG-Version: UML v2.4.1, 2011
- Standardisierte UML Profile z.B. siehe http://www.omg.org/spec/
- OMG Domain Specifications z.B. siehe http://www.omg.org/spec/
- Domain Specific Languages (DSLs)
- Rollenmodellierung

Übersicht UML-Diagramme

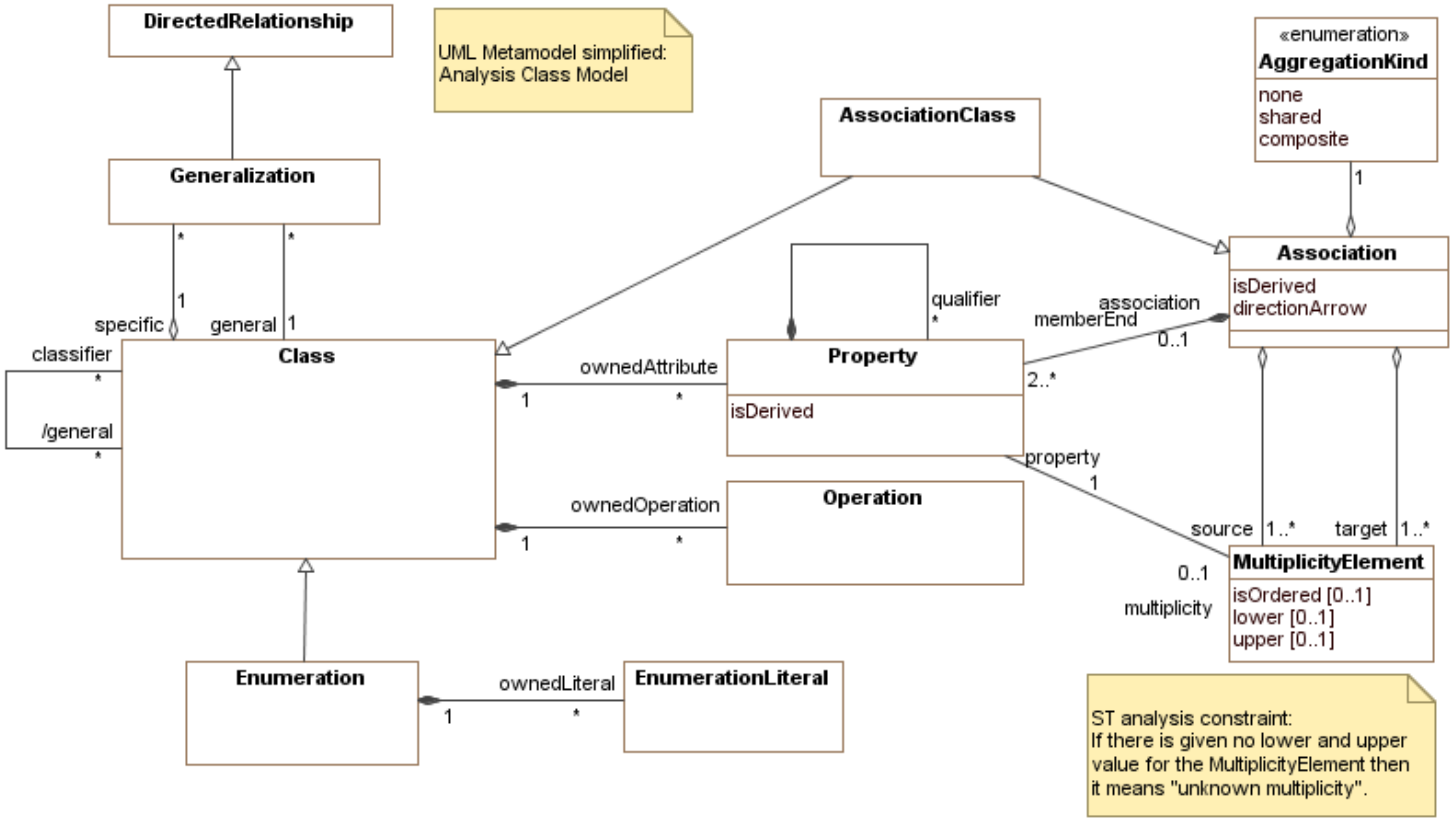


Objektorientierte Analyse (OOA)



aus Balzert /8/

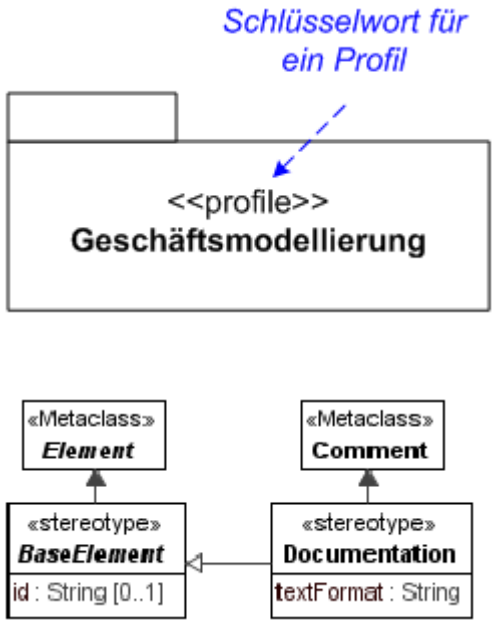
Metamodell für die OOA Klassendiagramme (ST I)



UML Profil

- Modellelement in der UML
- Erweiterungsmechanismus für das UML-Metamodell
- Profil = Spezialisierung von einem Paket
- besteht aus Stereotypen mit Tags (Attribute) und Constraints

- Wird für eine Anwendungsdomäne erstellt, z.B
- OMG UML Profile for BPMN 2 Processes
  - Semantisch äquivalent zum OMG BPMN 2 Metamodell



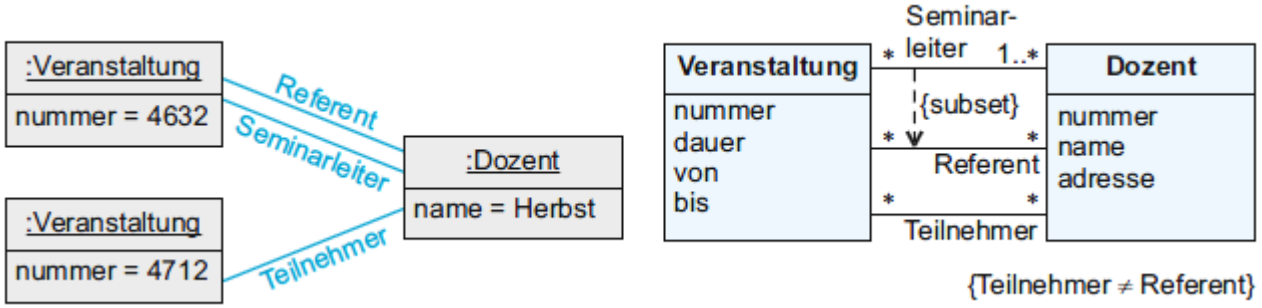
DSLs /4/

- DSLs im Vergleich zu Programmiersprachen (GPLs)
- DSLs im Requirement Engineering /4, S. 441 ff./
- Implementierung z.B. mit Xtext oder EMFText (Language Workbenches)

	more in GPLs	more in DSL
Domain Size	large and complex	smaller and well-defined
Designed by	guru or committee	a few engineers and domain experts
Language Size	large	small
Turing-completeness	almost always	often not
User Community	large, anonymous and widespread	small, accessible and local
In-language abstraction	sophisticated	limited
Lifespan	years to decades	months to years (driven by context)
Evolution	slow, often standardized	fast-paced
Incompatible Changes	almost impossible	feasible

Rollenmodellierung

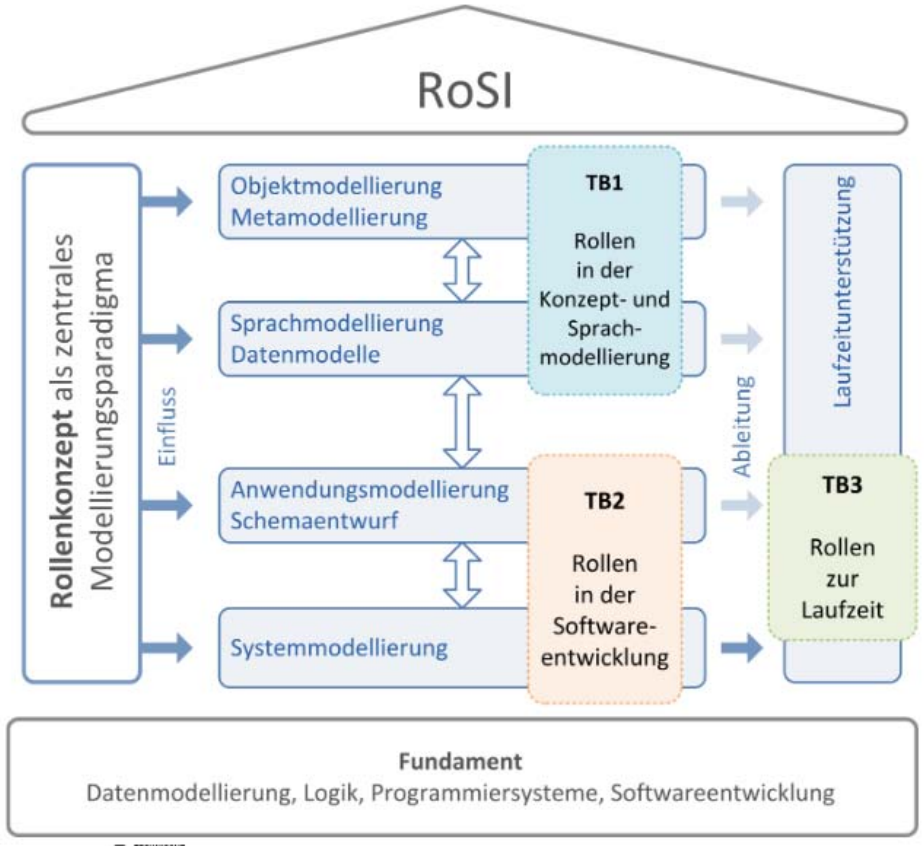
- Modellierung von Rollen seit Bachman, 1973
- Modellierung in der UML typischerweise mit Assoziationen (siehe ST I)
- Beispiel /8/:



- Rollen in der konzeptuellen Modellierung, theoretisch untersucht von
  - Steimann, 2000
  - Guizzardi, 2005

RoSI (Role-based Software Infrastructures)

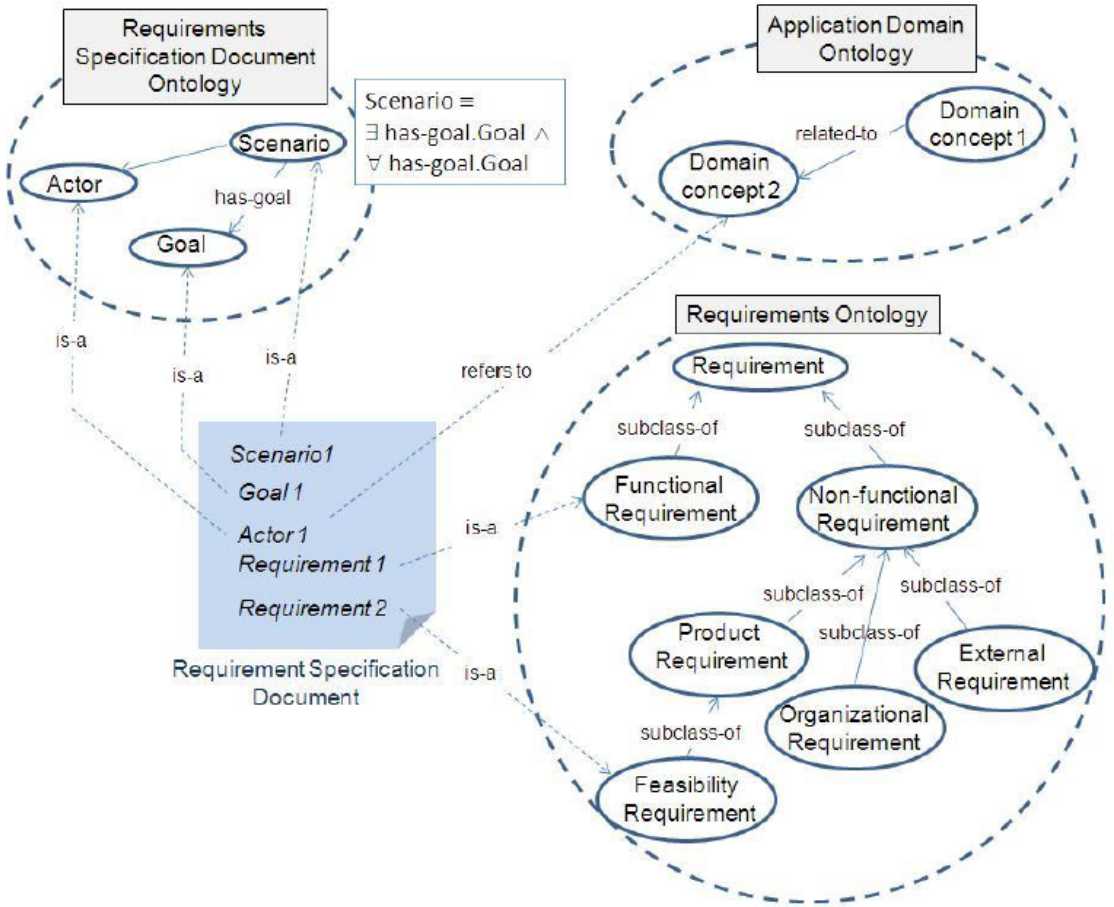
- Graduiertenkolleg an der Fakultät
- Forschungsfragen
  - Rollen und Zustand
  - Rollen und Identität



Ontologien

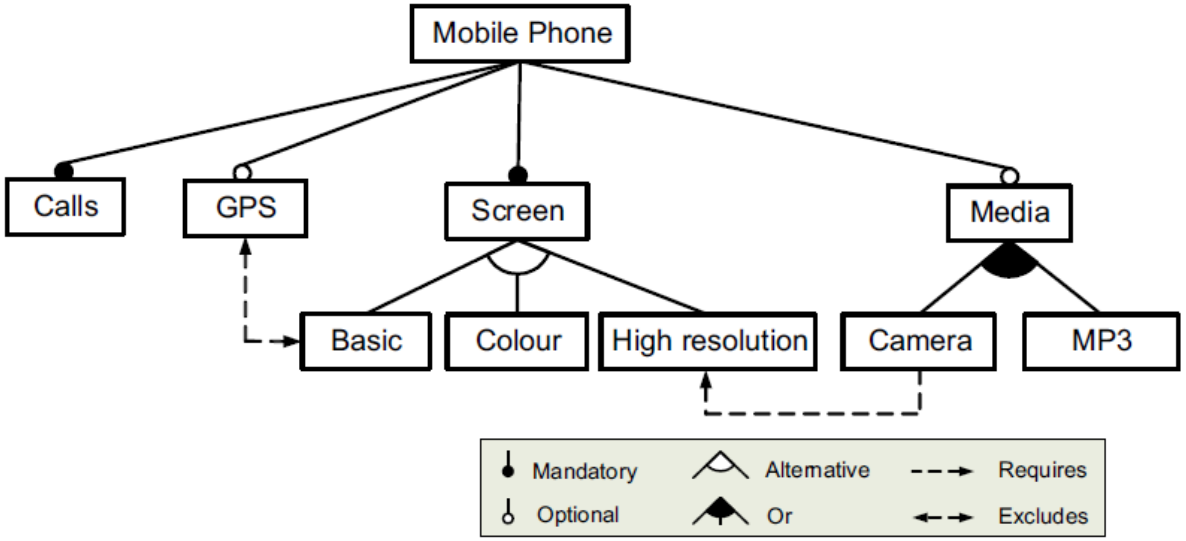
- Konzept aus der Philosophie
- Ontologien in der Informatik: KI-Forschung, Semantisches Web
- Definition einer Ontologie nach Gruber
  (http://tomgruber.org/writing/ontology-definition-2007.htm)
  - *In the context of computer and information sciences, an ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members). The definitions of the representational primitives include information about their meaning and constraints on their logically consistent application.*
- Standardisierung durch
  - W3C: Semantic Web (RDF, OWL, RIF, SPARQL, ...)
  - OMG: ODM (Ontology Definition Metamodel)
- Eigene Forschungsprojekte
  - REWERSE (Reasoning on the Web with Rules and Semantics) Network of Excellence
  - MOST (Marrying Ontology and Software Technology)

Ontologien im Requirements Engineering /5/

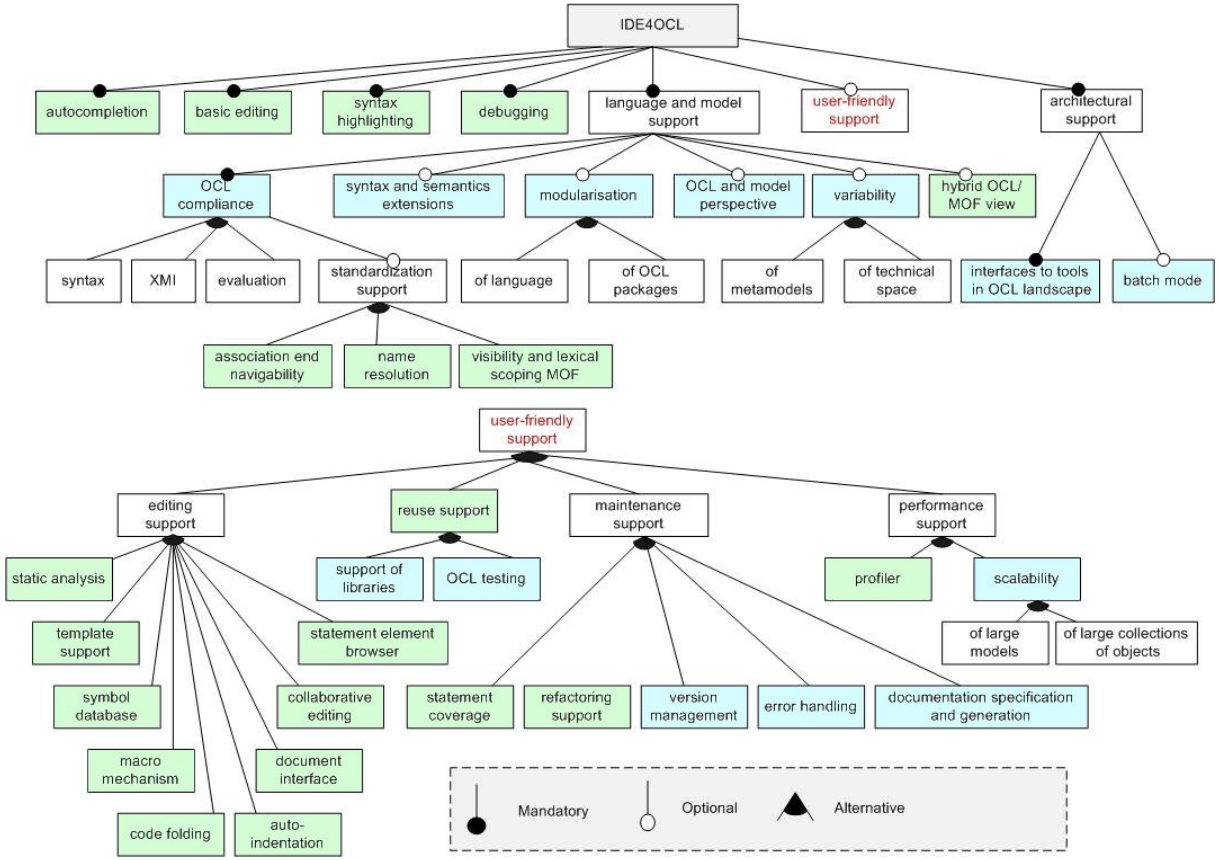


Featuremodellierung /6/

- Modellierung von Features und Beziehungen zwischen diesen aller möglichen Produkte einer **Software-Produktlinie (SPL)**

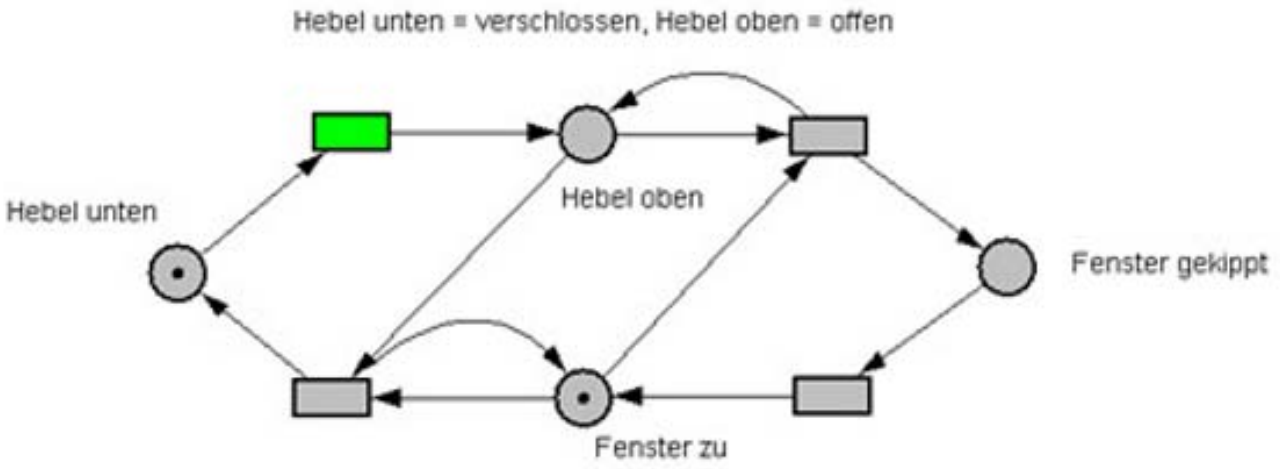


Beispiel IDE4OCL



Petrinetze

- Prozessmodellierung mit Petrinetzen (z.B. mit PIPE)
- Analyse, Modellbildung und Simulation von dynamischen Systemen mit nebenläufigen und nichtdeterministischen Vorgängen
- [Aufgabe]



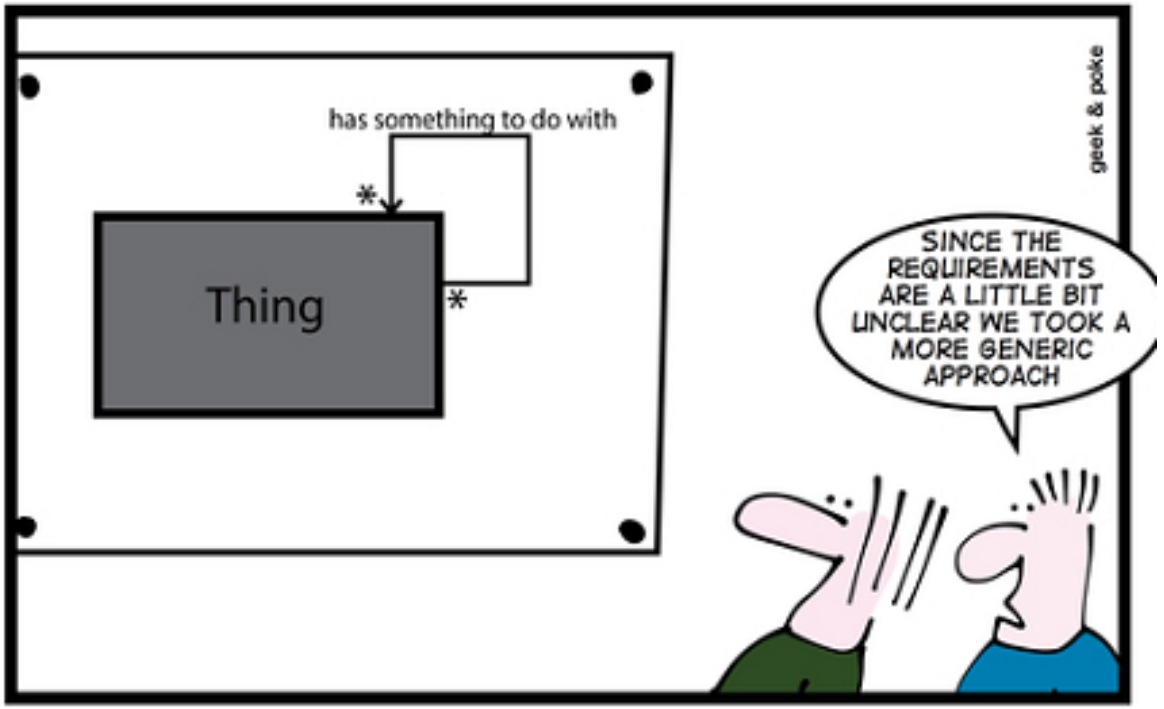
Formale Sprachen /7/

- Logik → Aussagenlogik, Prädikatenlogik 1.Ordnung
- Temporale Logik → LTL (Linear temporal Logic), CTL (Computation Tree Logic)
- Zustandsbasierte Spezifikationen → Z, VDM, B, Allooy, OCL
- Ereignisbasierte Spezifikationen → SCR, RSML, STATEMATE, LTS, Petrinetze
- Algebraische Spezifikationen → Abstrakte Datentypen
- Funktionen höherer Ordnung → HOL, PVS

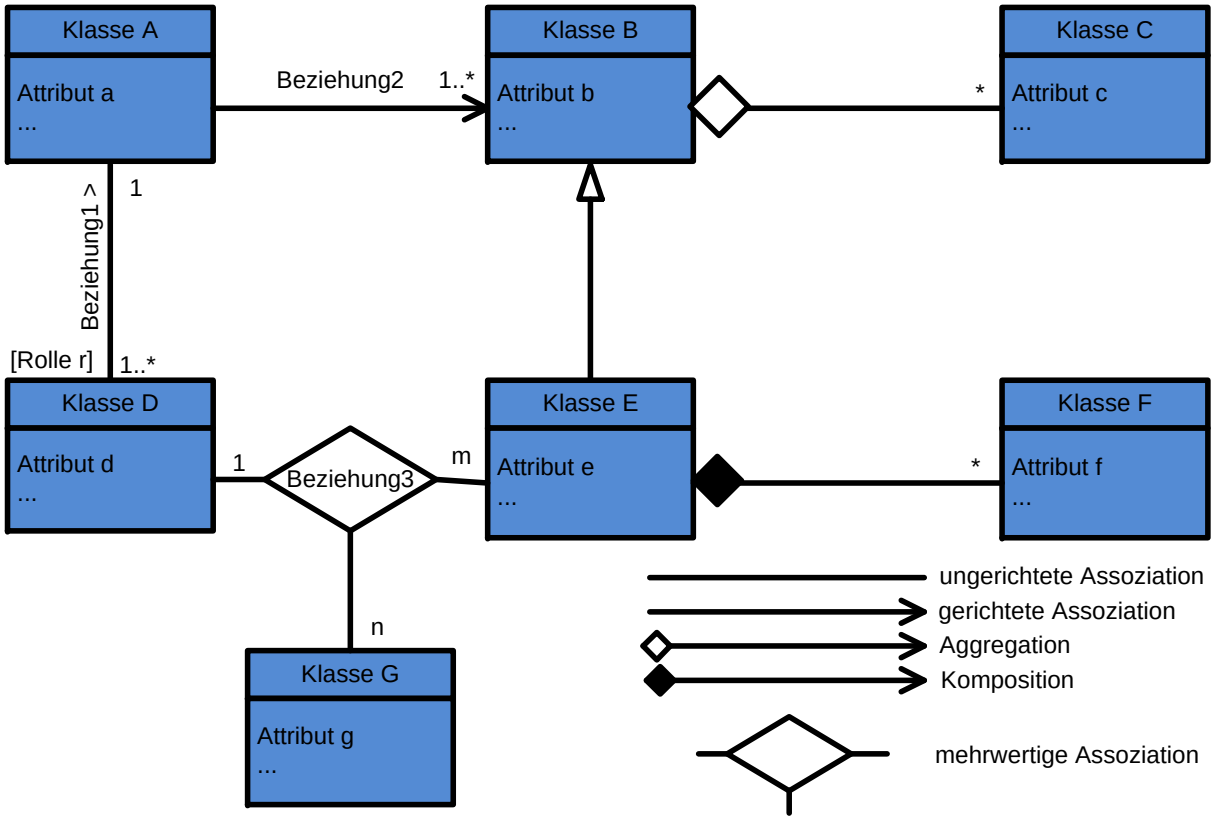
Themen ReqEng V

- Modellbildung
  - Modelle, Sichten, Notationen
  - Syntax, Semantik und Pragmatik
  - Formalisierung von Sprachen
- Modellierungstechniken
- Entwicklung des Anforderungsmodells
  - Sichten des Anforderungsmodells
  - Notationen und ihre Anwendung
  - Auswahl von Notationen
- Modellgetriebenes Requirements Engineering (MoDRE)

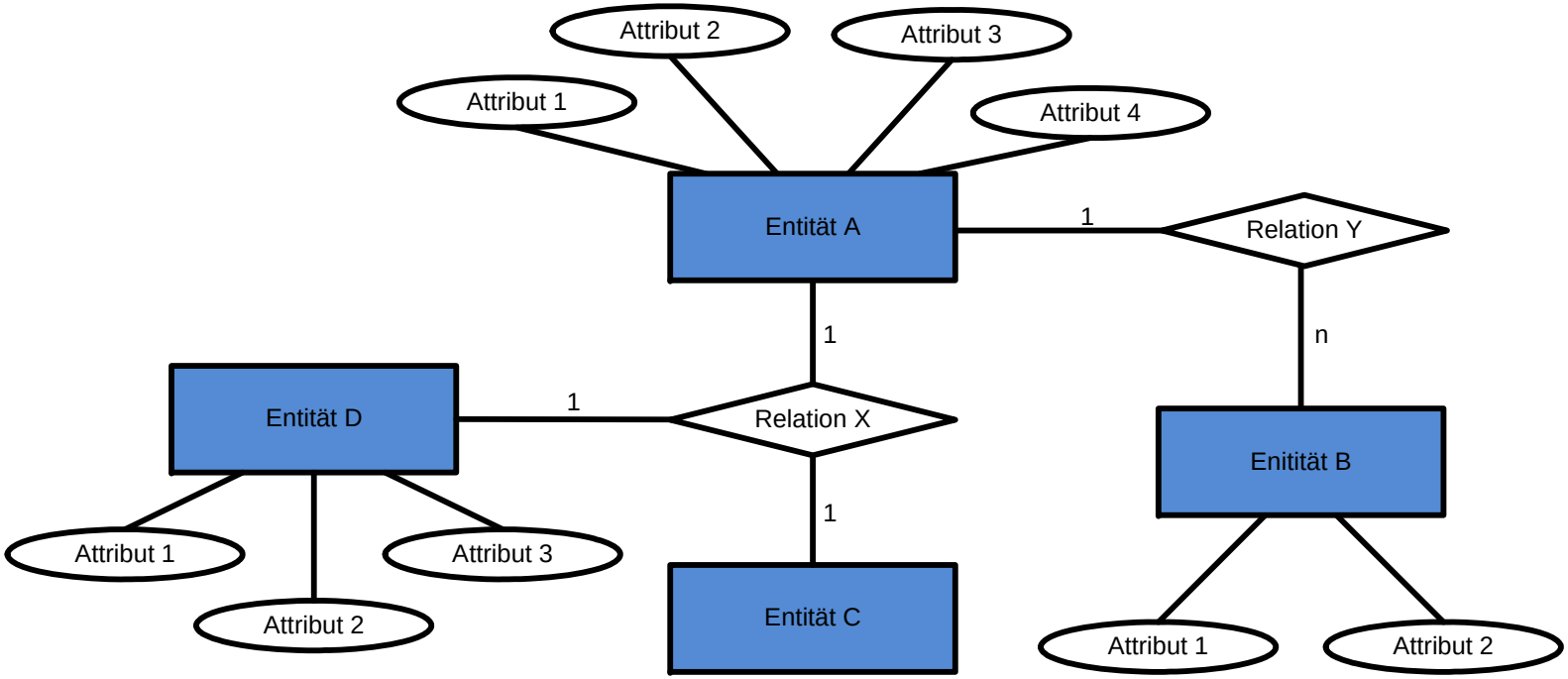
Struktursicht



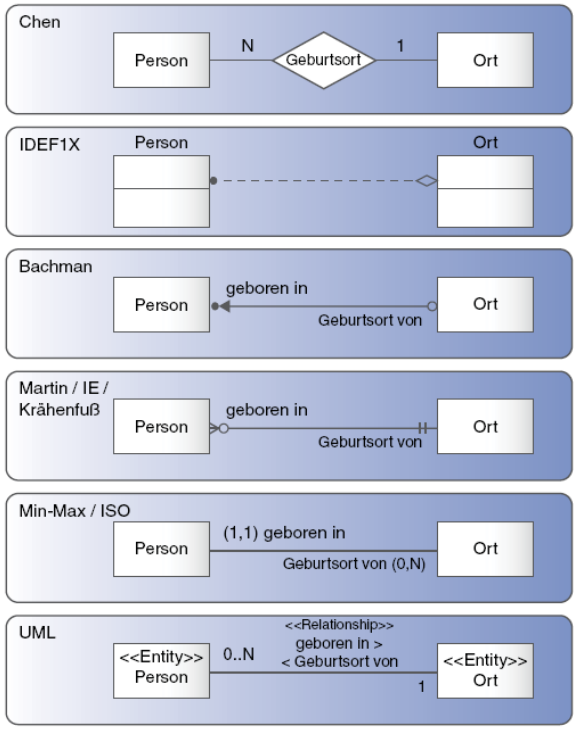
Struktursicht
UML Klassendiagramm



Struktursicht
ER Diagramm



Struktursicht
Informations- / Logische Datenmodellierung

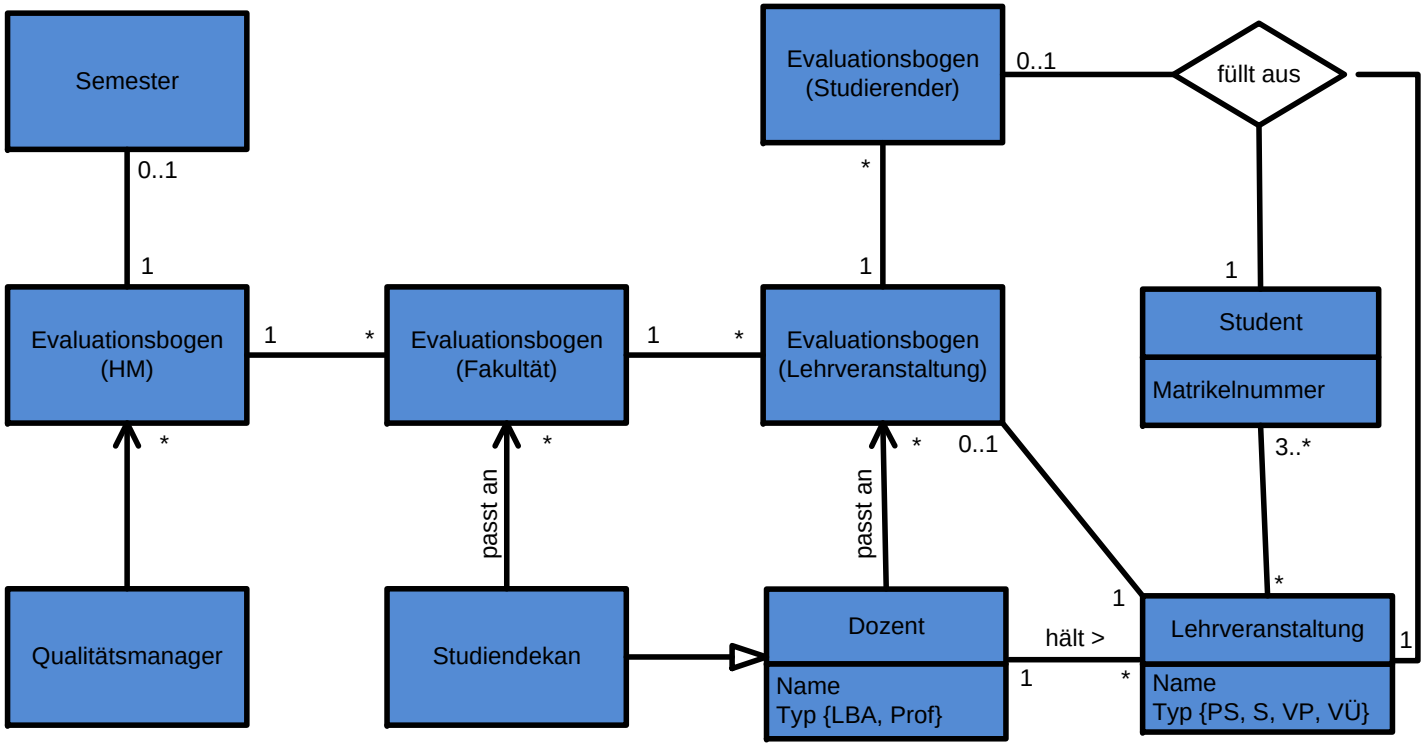


- seit Chen (1976, Erfinder des ER-Modells) viele verschiedene Notationen
- Informationsmodell
- Persistenzmodell
- häufig als relationales Modell
- oftmals aber auch XML Dokumente

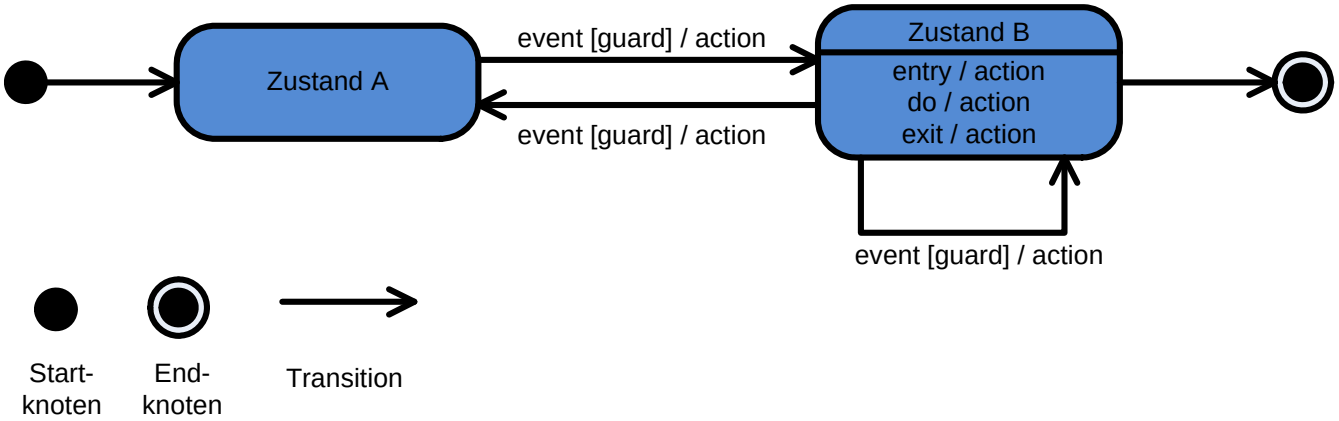
Anwendung im Requirements Engineering

- Entwicklung des **Domänenmodells**.
- Darstellung der strukturellen Zusammenhänge der Anwendungsdomäne:
  - Konzepte + Eigenschaften => Klassen + Attribute,
  - Beziehungen zwischen Konzepten => Assoziationen (Vererbung).
- Einschränkung der Notationselemente zur besseren Übersichtlichkeit: keine Methoden, keine Datentypen.
- Die Definition der Konzepte erfolgt im Glossar.
- Einsatz im RE: Analyse und strukturierte Darstellung des Problemraums.

Beispiel Evaluationssystem

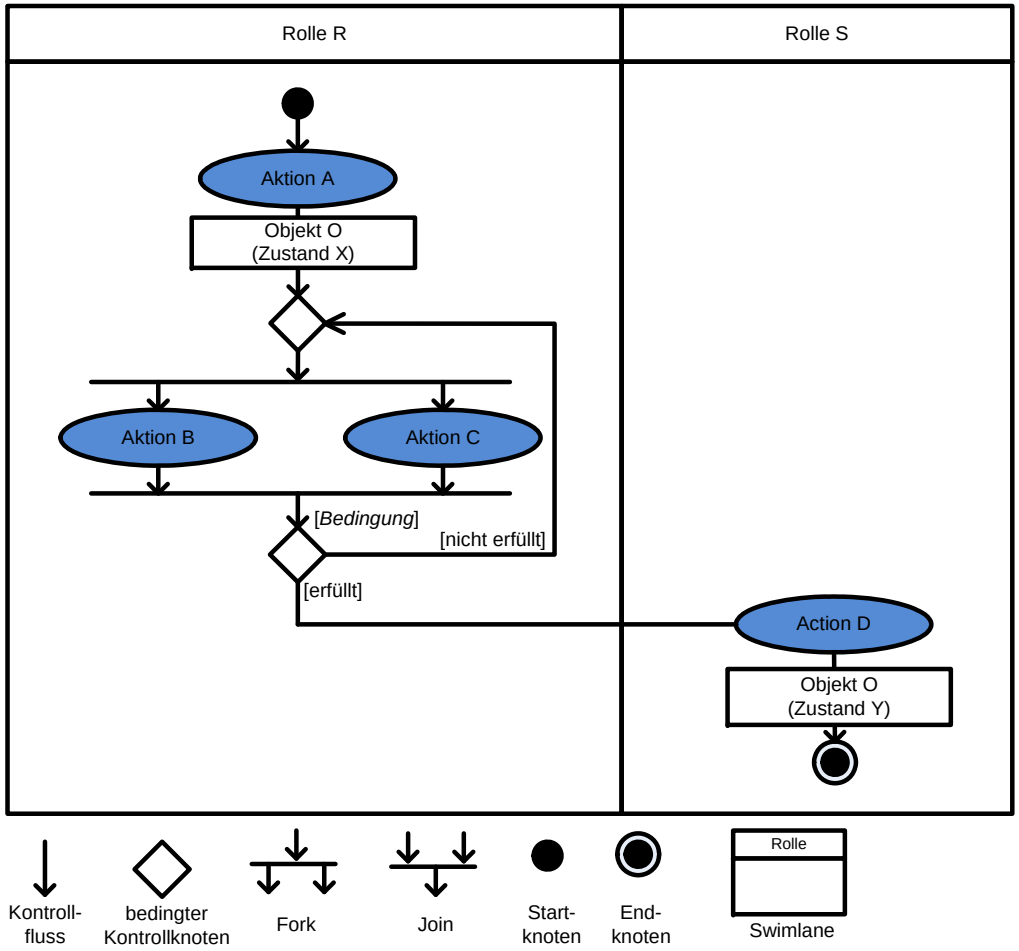


Verhaltenssicht – Notation UML Zustandsdiagramm



- Beschreibt das Verhalten komplexer fachlicher Objekte (Bestellung, Versicherungsantrag, etc.)
- Einsatz im RE: Analyse der Konzepte im Problemraum.

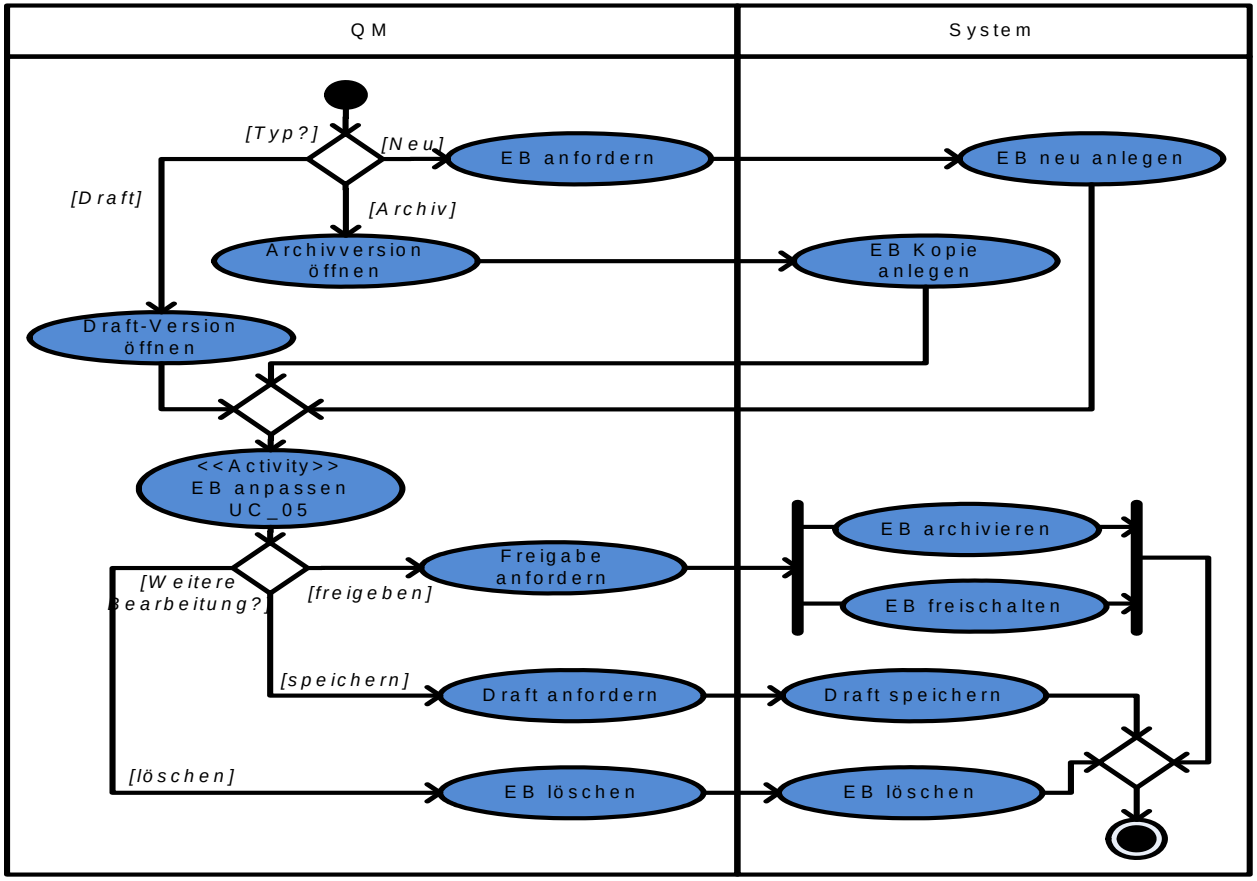
Schnittstellensicht Notation – Aktivitätsdiagramm



Anwendung im RE

- Modellierung von **Abläufen** auf unterschiedlichen Abstraktionsebenen.
  - Modellierung der Geschäftsprozesse,
  - Workflows (Use Case übergreifende Prozesse) an der Systemschnittstelle,
  - Abläufe von Use Cases (Standardablauf, Alternativabläufe, Fehlerabläufe),
  - Teilabläufe der Anwendungslogik (z.B. spezifische Algorithmen).

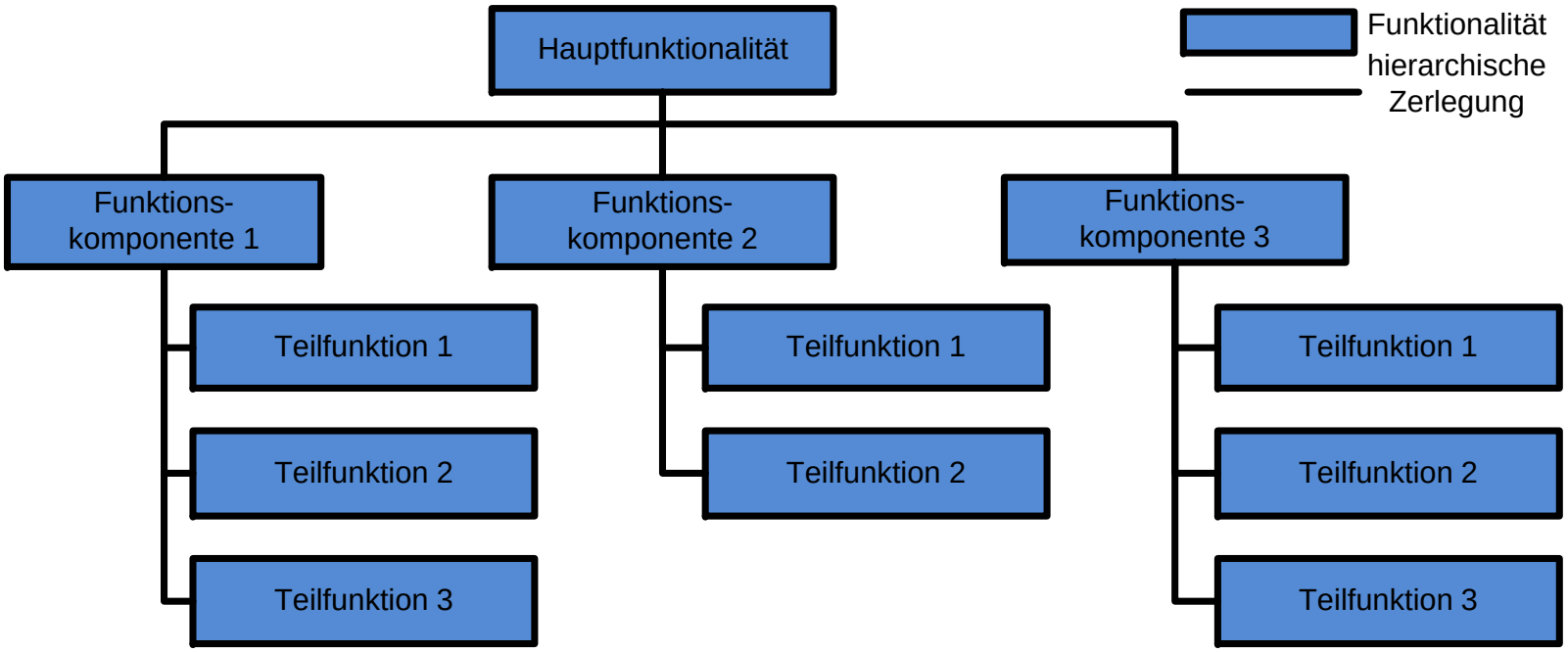
Beispiel Evaluationsystem



Beispiel Evaluationsystem

- Eva – Zustandsdiagramm /1, S. 210/
- Eva – Sequenzdiagramm /1, S. 213/

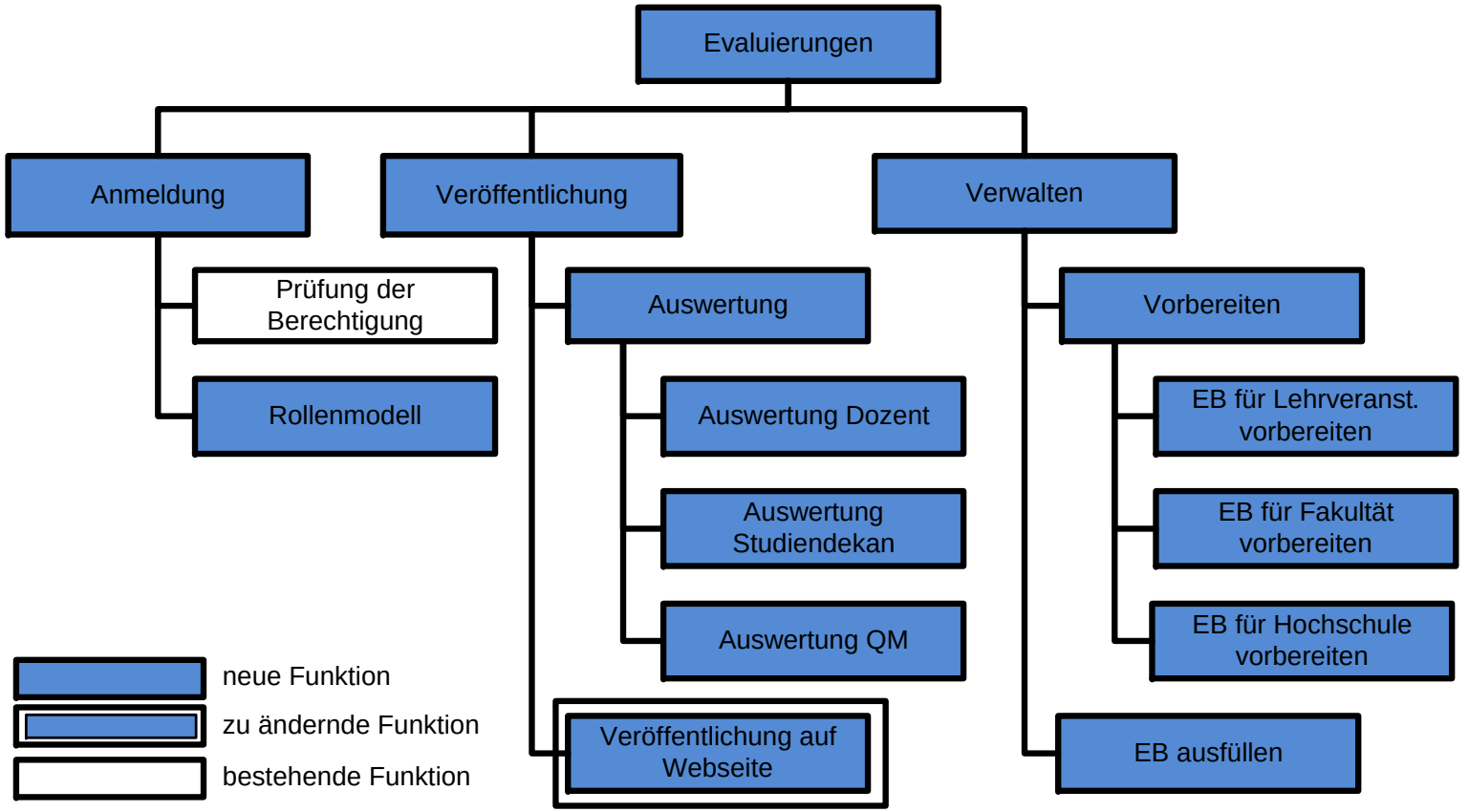
Funktionssicht - Notation Funktionsbaum



Anwendung im Requirements Engineering

- Notation der Strukturierten Analyse (zusammen mit Kontextdiagramm, Datenflussdiagramm, ER-Diagramm).
- Eignet sich hervorragend zur **funktionalen Dekomposition** des Systems.
- Leicht verständlich und nachvollziehbar.
- Informelle Notation, kann einfach um Sprachmittel ergänzt werden.

Funktionsbaum Beispiel



Sichten mit Notationen im Anforderungsmodell /1/

Sicht	Beschreibung	Notationen
Kontextsicht	Modelliert die Einbettung des Systems in seine Umgebung.	Kontextdiagramm
Funktionssicht	Modelliert die funktionale Einbettung des Systems in die existierende Systemlandschaft.	Funktionsbaum
Struktursicht (Domänenmodell, logische Datenmodelle)	Modelliert strukturelle Zusammenhänge der Anwendungsdomäne. Dies sind fachliche Konzepte mit ihren Eigenschaften und Beziehungen.	UML-Klassendiagramm ER-Diagramm
Verhaltenssicht	Modelliert das Verhalten fachlicher Objekte der Anwendungsdomäne.	UML-Zustandsdiagramm Petri-Netz Entscheidungstabellen
Schnittstellensicht	Modelliert das Schnittstellenverhalten des Systems.	Use-Case-Diagramm UML-Aktivitätsdiagramm UML-Sequenzdiagramm UML-Zustandsdiagramm

Auswahl von Notationen

Einflussfaktoren

- Akzeptanz durch Stakeholder
- Notationskenntnis bei den Stakeholdern
- Spezifikationslevel (Detailstufe)
- Art des Systems (technische, betriebswirtschaftliche, Informations-, … Systeme)
- Komplexität der zu beschreibenden Sachverhalte
- Technik zum Änderungsmanagement (Konsistenzerhaltung)
- Eindeutigkeit

Auswahlempfehlungen in /2/

Themen ReqEng V

- Modellbildung
  - Modelle, Sichten, Notationen
  - Syntax, Semantik und Pragmatik
  - Formalisierung von Sprachen
- Modellierungstechniken
- Entwicklung des Anforderungsmodells
  - Sichten des Anforderungsmodells
  - Notationen und ihre Anwendung
  - Auswahl von Notationen
- Modellgetriebenes Requirements Engineering (MoDRE)

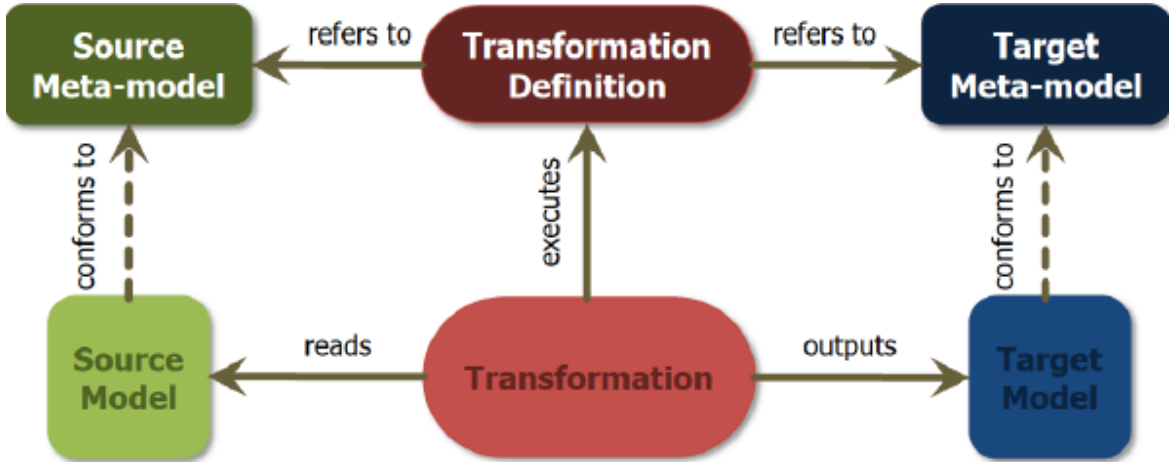
Modellgetriebenes Requirements Engineering (MoDRE) /9/

MoDRE = **Modellbasiertes** Requirements Engineering + **Transformationen**

- Model-to-model transformation
- Model-to-text transformation
- Text-to-Model transformation
- In-place model transformation

Transformationen:

- Horizontal
- Vertikal
- Zwischen Paradigmen (Technikräumen)



Vorteile MoDRE
(noch großes Forschungsgebiet)

- Konsistenzsicherung zwischen verschiedenen Arten von Analysemodellen
- Sicherung von WFRs in Modellen
- Ableitung von Architekturmodellen aus Anforderungen
- Unterstützung von Trennung von Belangen (Separation of Concerns), Wiederverwendung und Komposition von Belangen
- Verifikation von Qualitätsanforderungen an Analysemodelle
- Unterstützung der Evolution von Analysemodellen (change impact analysis, change propagation)

Literatur für ReqEng V

(1) Ulrike Hammerschall, Gerd Beneken: Software Requirements. PEARSON, 2013 (Primärliteratur)

(2) Chris Rupp und die SOPHISTen: Requirements-Engineering und -Management. HANSER, 2009 (Sekundärliteratur)

(3) Janis A. Bubenko: From Information Algebra to Enterprise Modelling and Ontologies – a Historical Perspective on Modelling for Information Systems. In: John Krogstie et al: Conceptual Modelling in Information Systems Modelling. Springer, 2007

(4) Markus Voelter et al: DSL Engineering:Designing, Implementing and Using Domain-Specific Languages. dslbook.org, 2010 – 2013 (Donationware)

(5) Verónica Castañeda et al: The Use of Ontologies in Requirements Engineering. Global Journal of Researches in Engineering. 10(2010)6

(6) David Benavides et al: Automated analysis of feature models 20 years later: A literature review. Elsevier, 2010, doi:10.1016/j.is.2010.01.001

Literatur für ReqEng V

(7) Axel van Lamsweerde: Requirements Engineering. From System Goals to UML Models to Software Specifications. John Wiley, 2009

(8) Balzert, Helmut. Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering, 3. Auflage. Spektrum Akademischer Verlag, 2009.

(9) João Araújo, Ana Moreira, Gunter Mussbacher, Pablo Sanchez: Model-Driven Requirements Engineering. Tutorial , MODELS 2014, Valencia, 2014.

Weiterführende Literatur für ReqEng V

- Booch, Gradys. Object-Oriented Analysis and Design with Applications. Addison-Wesley Professional, 1993.
- Chen, Peter. „The Entity-Relationship Model-Toward a Unified View of Data.“ ACM Transactions on Database Systems ACM-Press ISSN 0362-5915 1976: S. 9–36.
- Coad, Peter und Edward Yourdon. Objekt-orientierte Analyse. München: Prentice Hall, 1994.
- DeMarco, Tom. Structured analysis and system specification. Prentice Hall International, 1979.
- ISO/IEC 13568. „Information Technology - Z Formal Specification Notation – Syntax, Type System and Semantics.“ 2002.

Weiterführende Literatur für ReqEng V

- Jacobson, Ivar, Magnus Christerson und Patrik Jonsson. Object Oriented Software Engineering - A Use Case Driven Approach. Addison-Wesley, 1992.
- Jürjens, Jan. „A UML statecharts semantics with message-passing.“ Proceedings of the 2002 ACM symposium on Applied computing 2002: 1009--1013.
- Kecher, Christoph. UML 2.0: Das umfassende Handbuch, 2. Auflage. Galileo Computing, 2006.
- OMG. „Unified Modeling Language (UML), Version 2.4.1.“ http://www.uml.org: Object Management Group, August 2011.
- Rumbaugh, James. Object-Oriented Modeling and Design. Prentice Hall, 1990.
- Yourdon, Edward. Modern Structured Analysis. Prentice Hall, 1988.