UML 2 – Fakten, Fallstricke und konkrete Fallstudien

Chris Rupp

SOPHIST GROUP

chris.rupp@sophist.de

Prof. Mario Jeckle

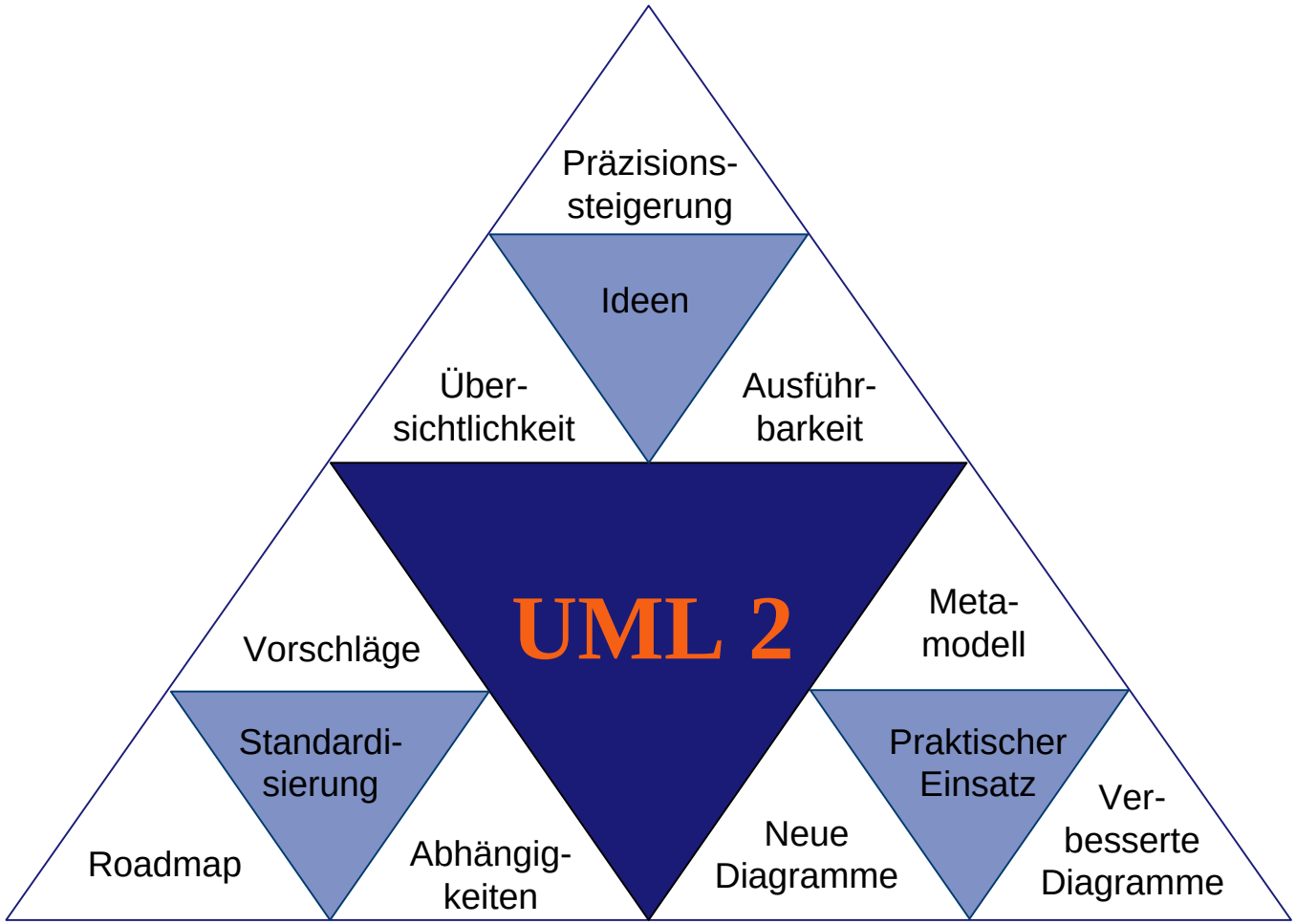
Fachhochschule Furtwangen,
Vertreter der DaimlerChrysler AG in der OMG

mario@jeckle.de

Es gibt Menschen, die, wenn sie das Licht am Ende des Tunnels sehen, ein neues Stück Tunnel kaufen.

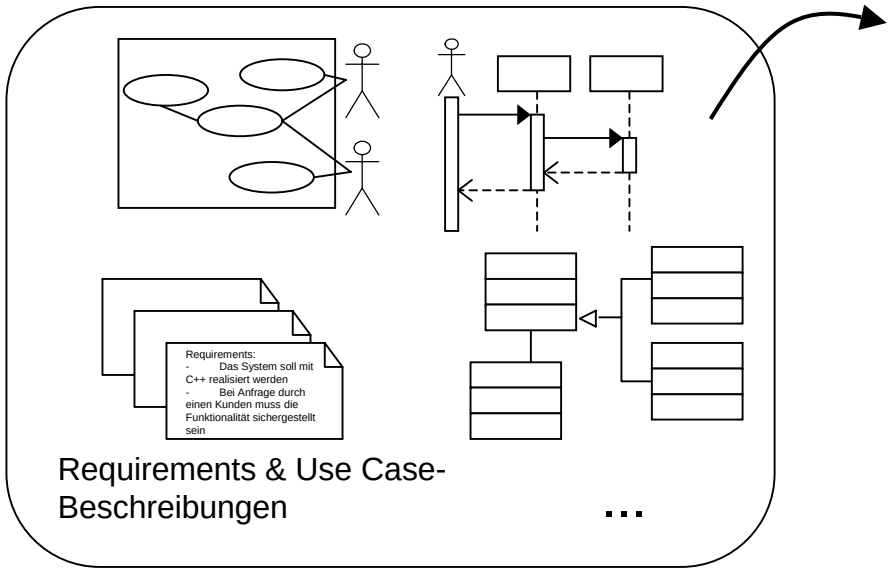
Johannes Rau

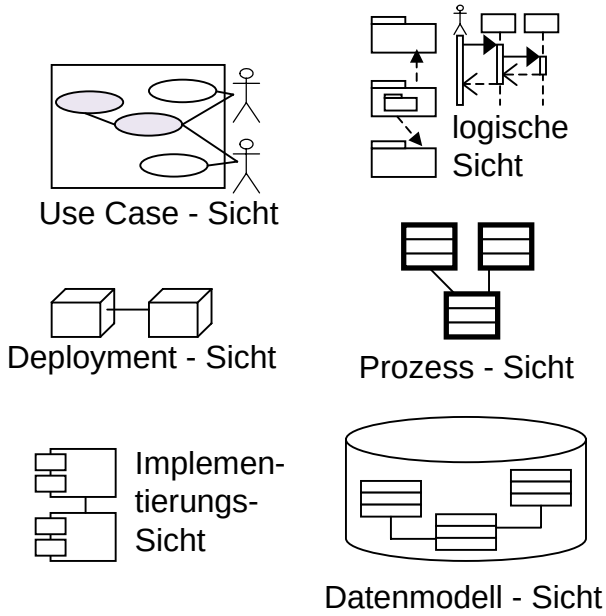
Darum geht´s


Präzisions-
steigerung
Ideen
Über-
sichtlichkeit
Ausführ-
barkeit
UML 2
Vorschläge
Meta-
modell
Standardi-
sierung
Praktischer
Einsatz
Roadmap
Abhängig-
keiten
Neue
Diagramme
Ver-
besserte
Diagramme

Was ist die UML?

Wunsch des Kunden

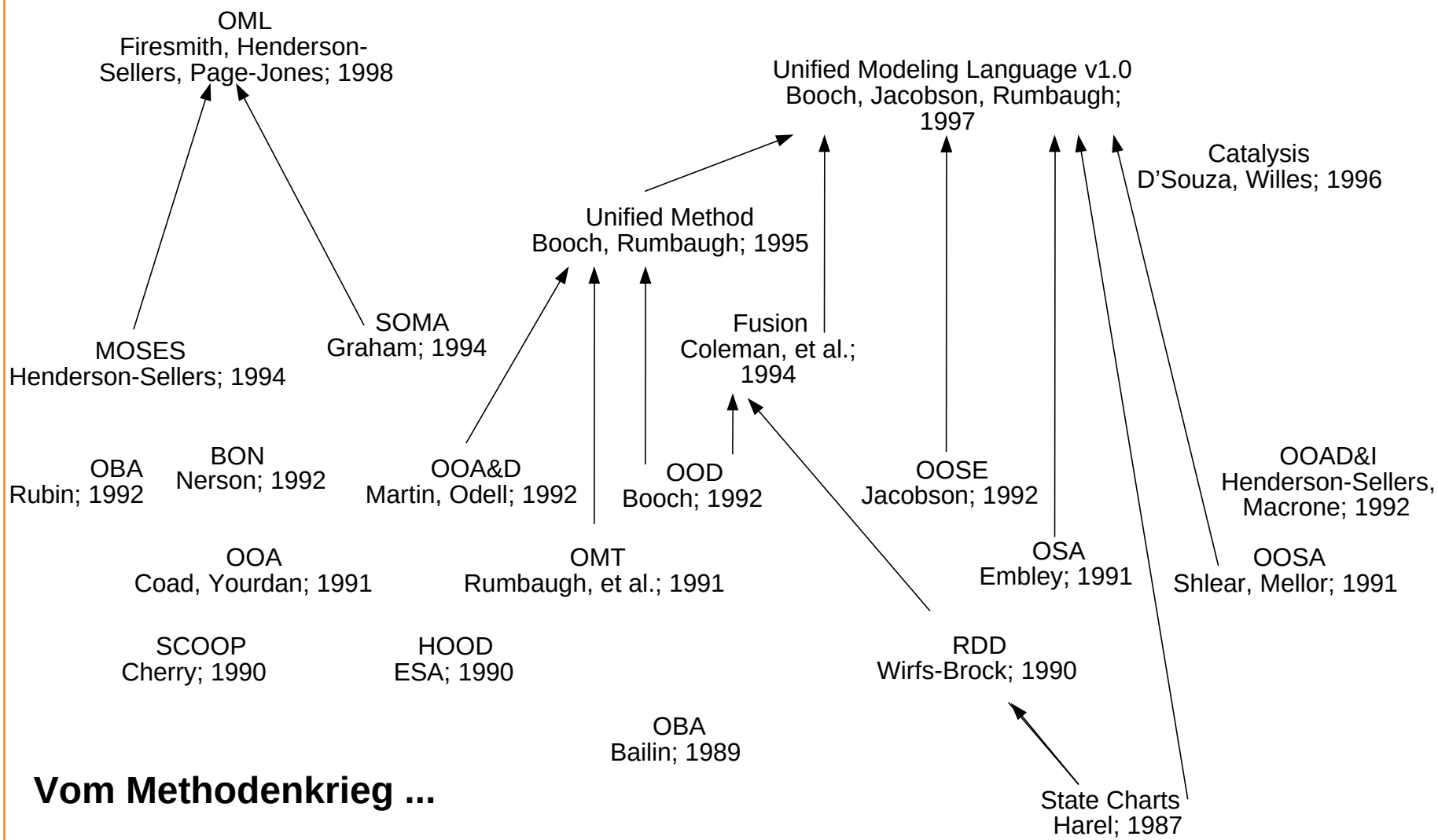




- Notation für Modell des Systems
- zeigt statische und dynamische Aspekte
- Nicht jede Sicht in jedem System sinnvoll

UML ...

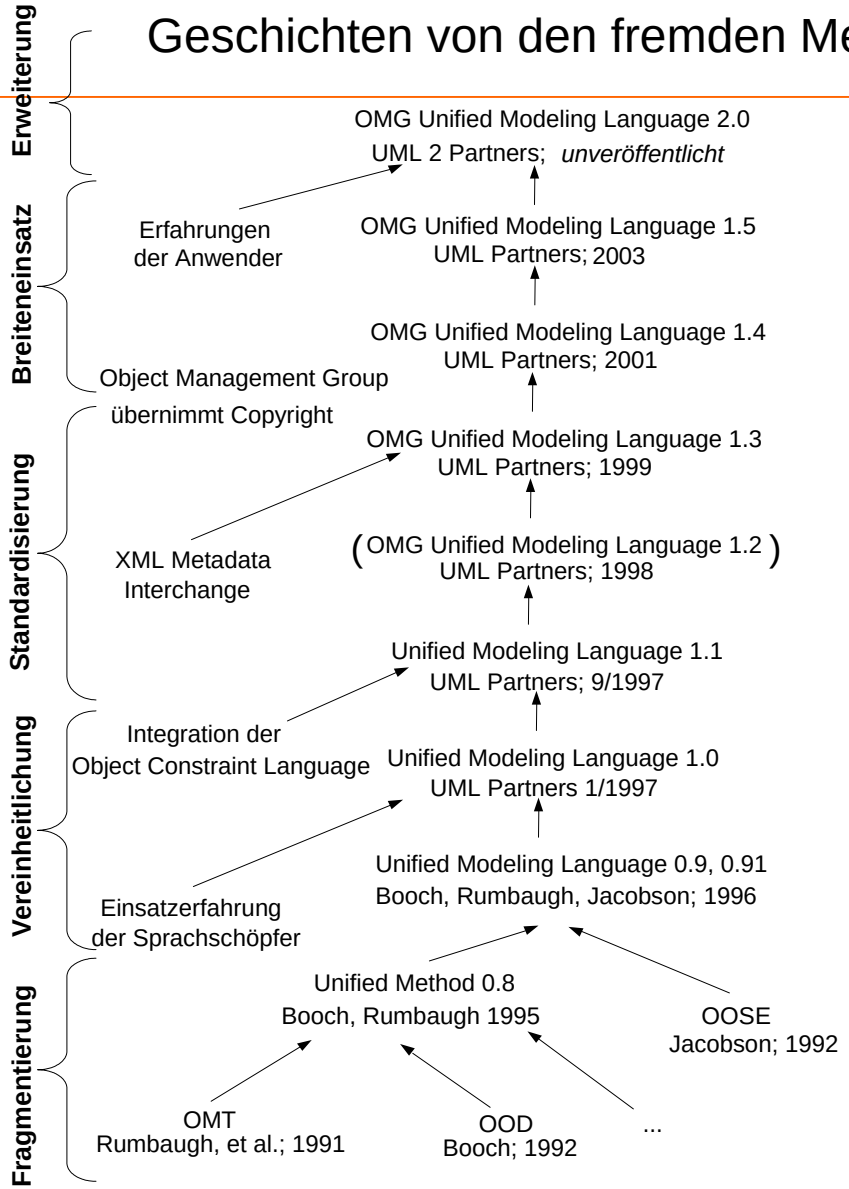
Geschichten von den fremden Meeren der Standardisierung



Vom Methodenkrieg ...

UML ...

Geschichten von den fremden Meeren der Standardisierung



Viele arbeiten für alle

Einige arbeiten für andere

Wenige arbeiten für einige

... zum Standardisierungskrieg

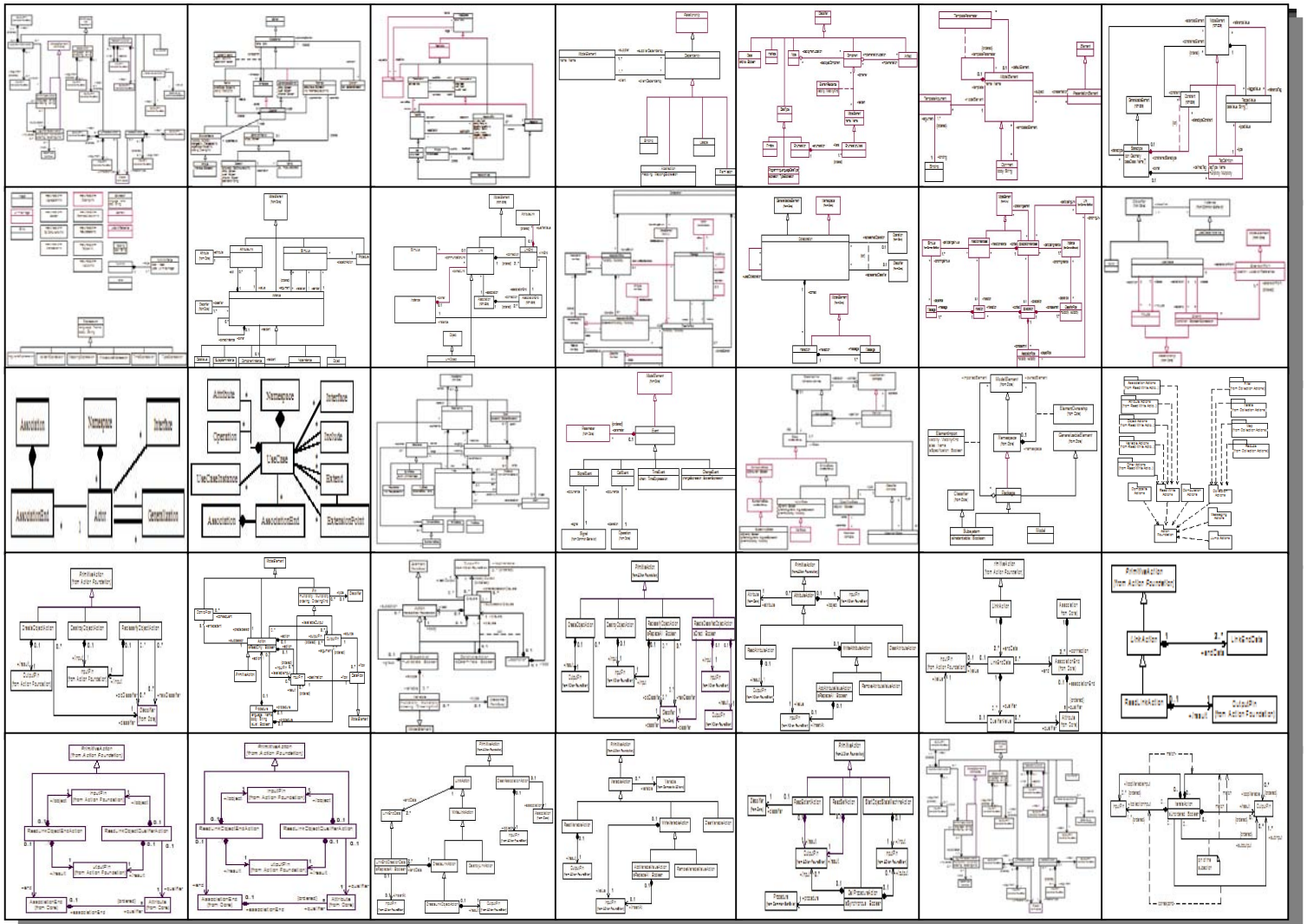
UML 2

... Warum eine neue Version?

- **Evolution**
  - Der Markt hat sich bewegt...
    - Neue Programmiersprachen (z.B. C#, Python, PHP)
    - Neue Anwendungsdomänen (z.B. Serverprogrammierung, Echtzeitanwendungen)
- **Erfahrung**
  - Für einige Einsatzgebiete bietet UML v1.x ...
    - Manchmal zu wenig Konstrukte
    - Manchmal zu viele
    - Machmal so viele, dass die sinnvolle Auswahl schwerfällt
- **Eliminierung**
  - Einige Programmiersprachen verschwinden (z. B. C++)
  - Einige früher als modellierungsnah eingestufte Konzepte entwickeln sich inzwischen getrennt von UML weiter (z. B. Entwicklungsprozesse, Codegenerierung)

UML ...

Ein Leiden am Second System Syndrom



UML 2

Die Ziele

- **Übersichtlichkeit**
  - Weniger graphische Modellkonstrukte
  - Weniger Basiskonzepte
  - Wiederverwendung von Basiskonzepten
- **Präzisionssteigerung**
  - Reformulierung des Meta-Modells
  - Weitestgehende OCL-Verwendung
  - Unveränderte Wiederverwendung von Basiskonstrukten soweit sinnvoll möglich
- **Ausführbarkeit**
  - Erweiterte Zustandsmaschinen
  - Stärkere Beziehungen zwischen statischen und dynamischen Diagrammen
  - Integration erprobter Konzepte außerhalb der UML

UML 2

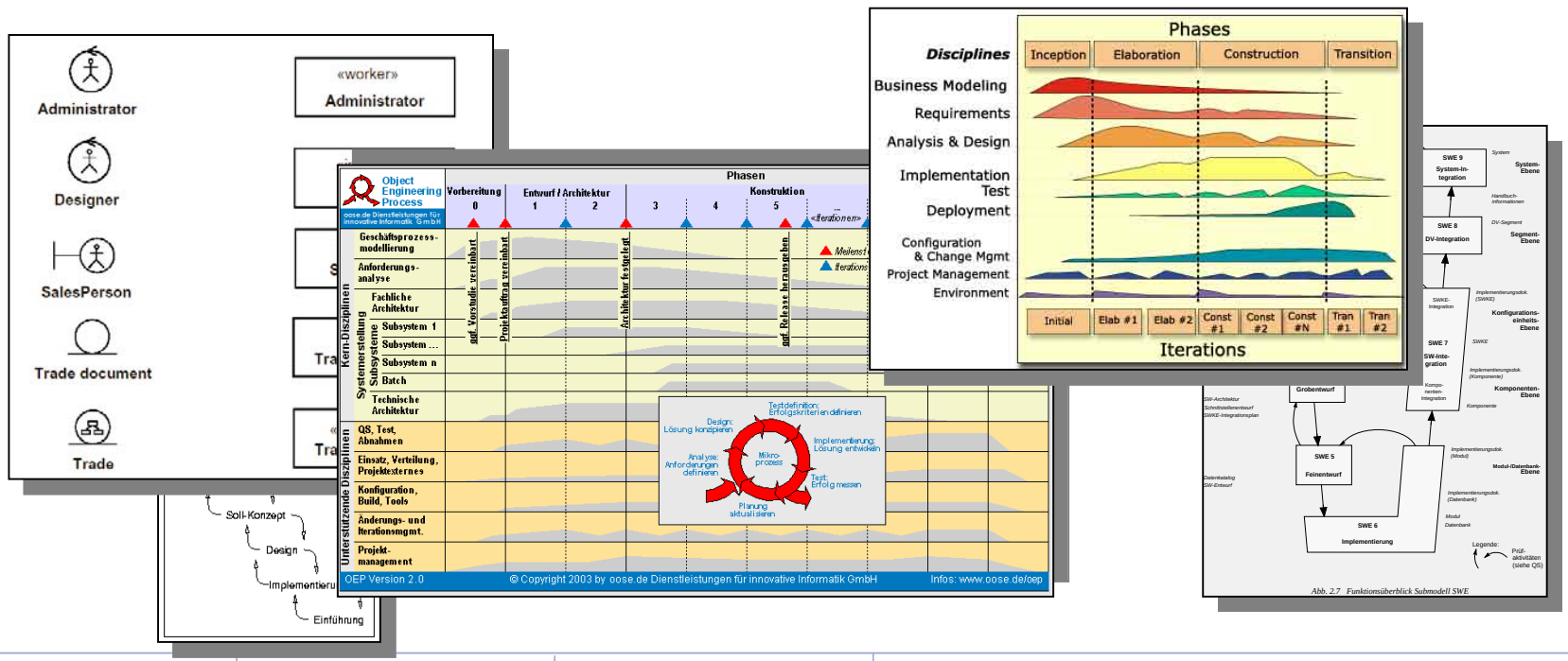
Verrentung existierender Modellelemente

1. Durch UML-Werkzeuge nicht implementierte Sprachanteile
2. Durch OO-Methoden unberücksichtigte Sprachelemente
3. Programmiersprachen-spezifische Sprachelemente
4. Inpräzise UML-Sprachelemente

UML 2

Verrentung existierender Modellelemente

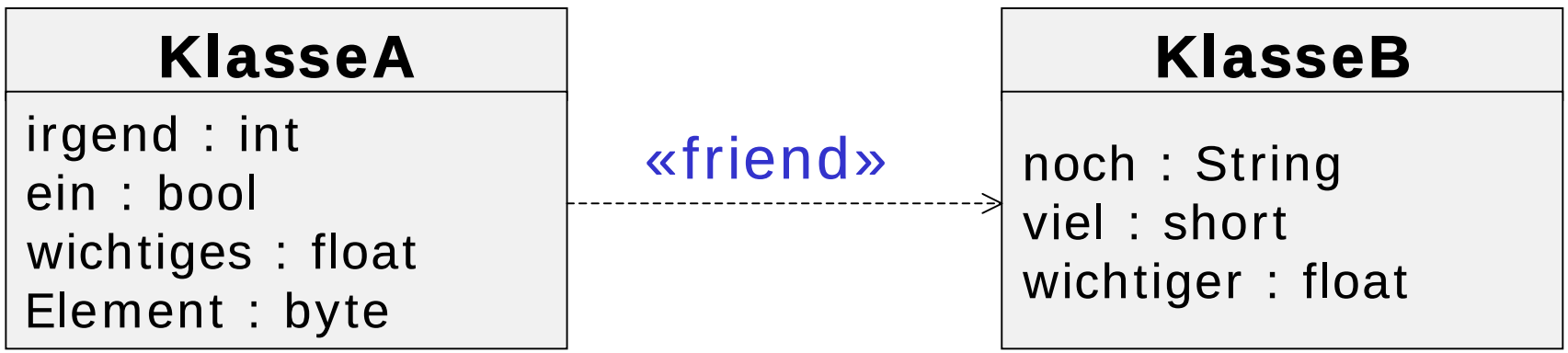
1. Durch UML-Werkzeuge nicht implementierte Sprachanteile
2. Durch OO-Methoden unberücksichtigte Sprachelemente
3. Programmiersprachen-spezifische Sprachelemente
4. Inpräzise UML-Sprachelemente



UML 2

Verrentung existierender Modellelemente

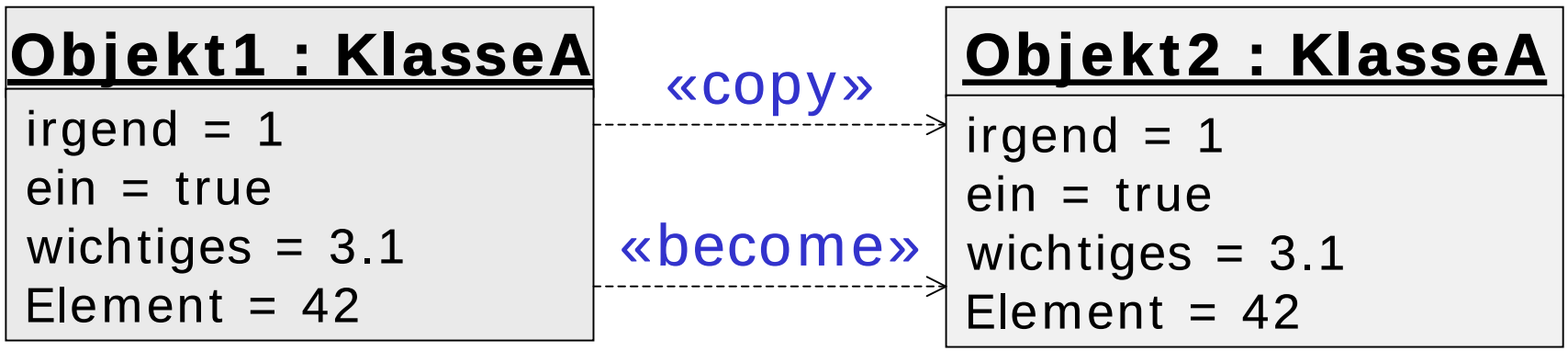
1. Durch UML-Werkzeuge nicht implementierte Sprachanteile
2. Durch OO-Methoden unberücksichtigte Sprachelemente
3. Programmiersprachen-spezifische Sprachelemente
4. Inpräzise UML-Sprachelemente



UML 2

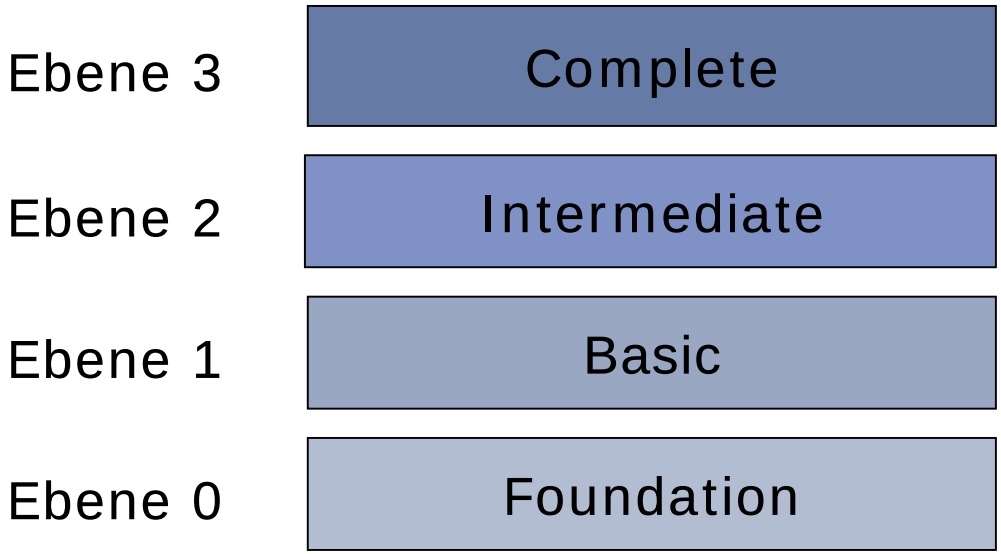
Verrentung existierender Modellelemente

1. Durch UML-Werkzeuge nicht implementierte Sprachanteile
2. Durch OO-Methoden unberücksichtigte Sprachelemente
3. Programmiersprachen-spezifische Sprachelemente
4. Inpräzise UML-Sprachelemente

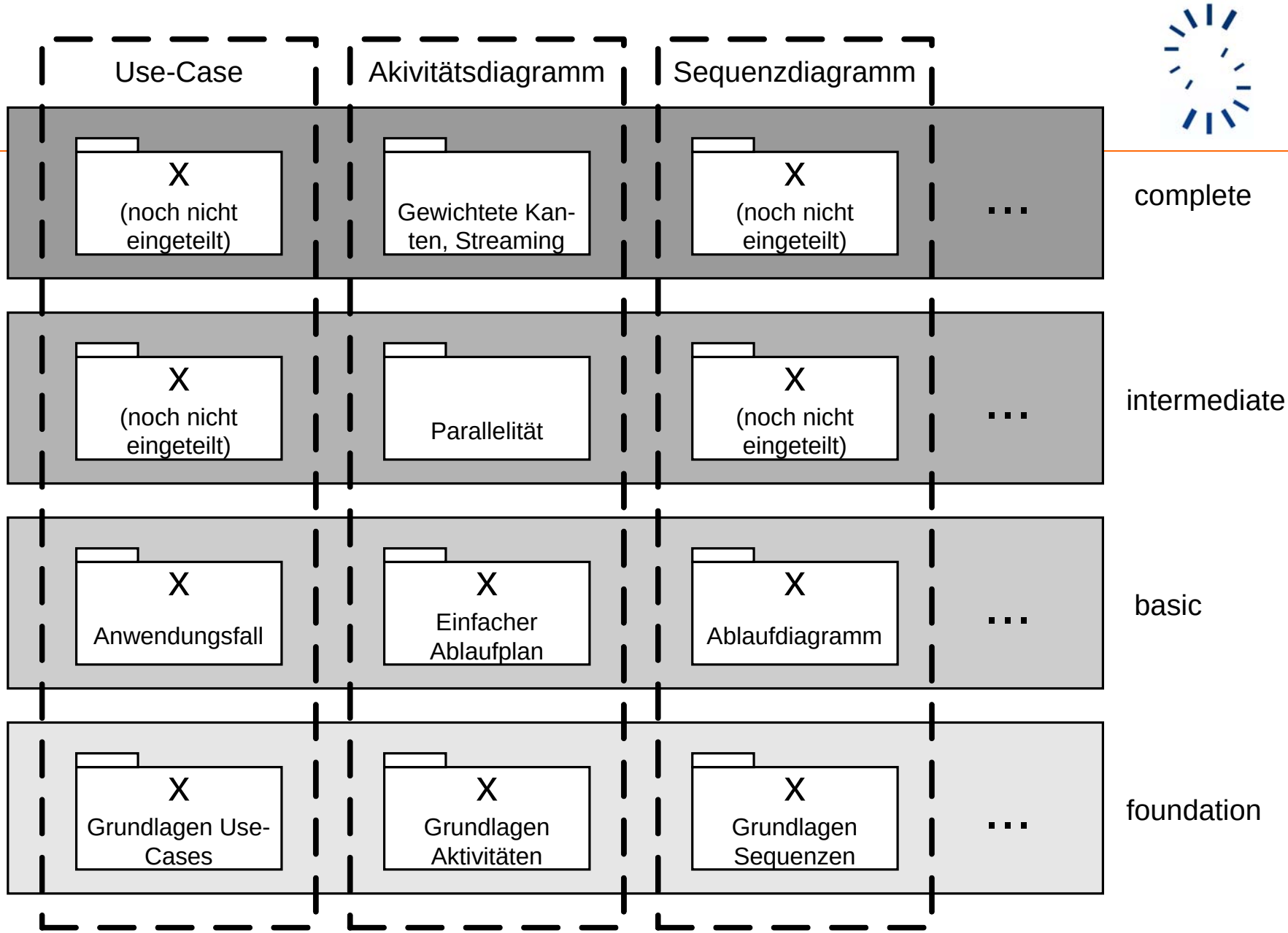


UML 2

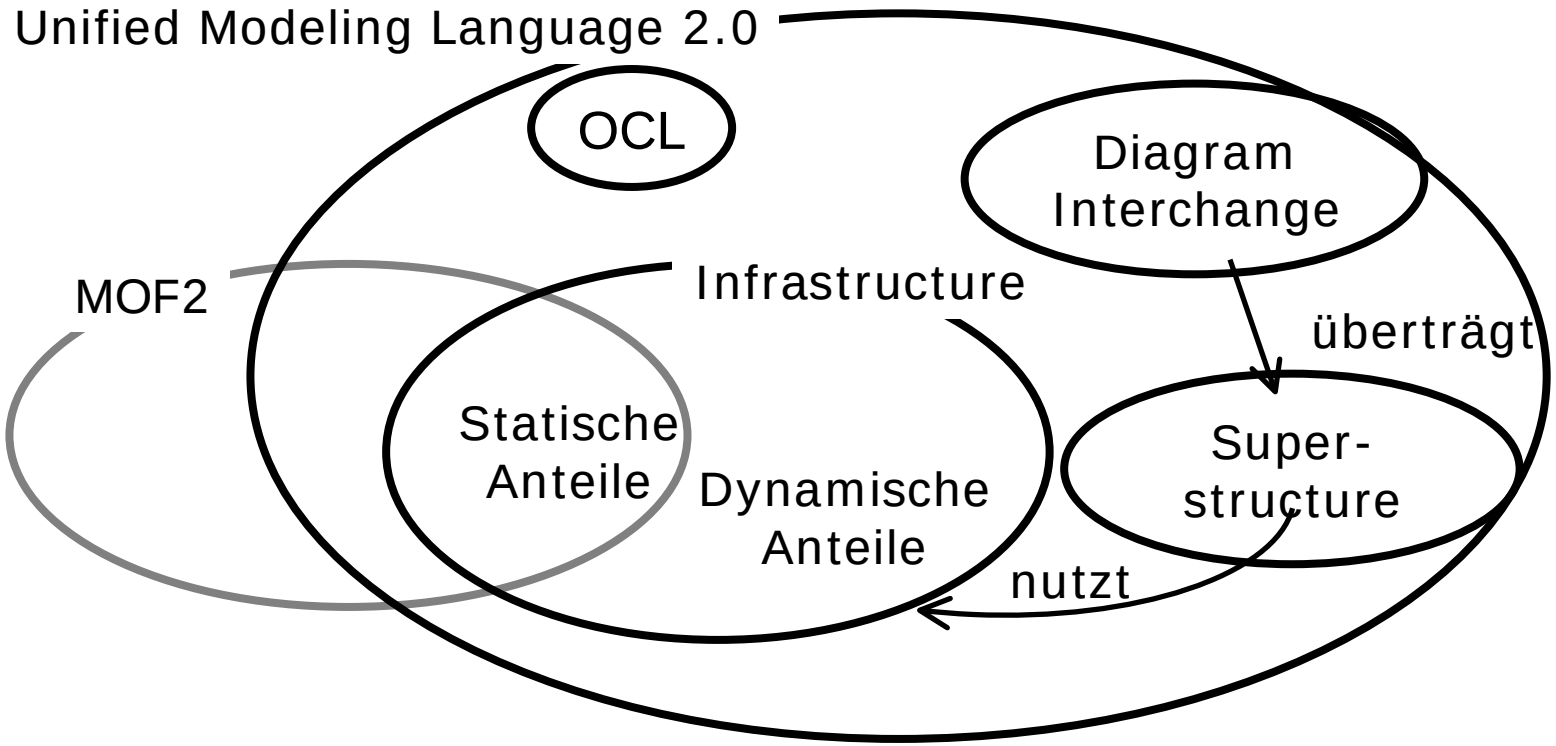
Neu: UML Schichten



- Die Idee entstammt der SQL-Standardisierung
- Operationalisiert den Begriff der UML-Unterstützung
- Auch weniger UML ist immer noch UML


Use-Case
Akivitätsdiagramm
Sequenzdiagramm
X
(noch nicht eingeteilt)
Gewichtete Kan-ten, Streaming
X
(noch nicht eingeteilt)
...
complete
X
(noch nicht eingeteilt)
Parallelität
X
(noch nicht eingeteilt)
...
intermediate
X
Anwendungsfall
X
Einfacher Ablaufplan
X
Ablaufdiagramm
...
basic
X
Grundlagen Use-Cases
X
Grundlagen Aktivitäten
X
Grundlagen Sequenzen
...
foundation

Struktur und Einbettung von UML 2



- UML ist nicht mehr eine monolithische Sprache
- Vier seperate Entwicklungsgruppen werden vier seperate Weiterentwicklungen mit starken inneren Zusammenhang erzeugen

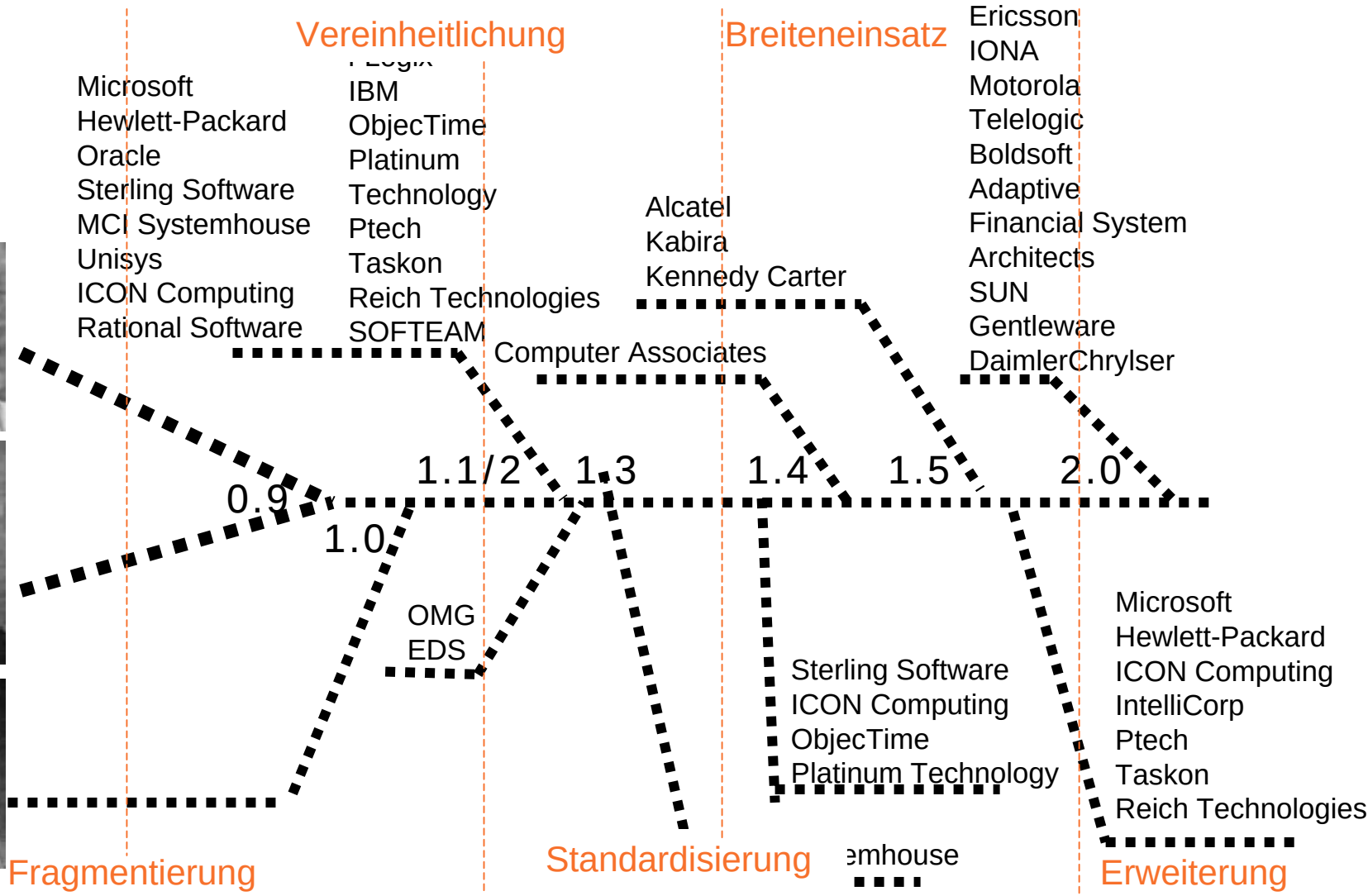
Struktur und Einbettung von UML 2

- Verschiedene Weiterentwicklungsvorschläge:
  - **Infrastructure:** 36 Lettern of Intents (LOIs);
    5 Einreichungen durch 28 Firmen
  - **Superstructure:** 37 LOIs;
    5 Einreichungen durch 28 Firmen
  - **OCL:** 30 LOIs; 4 Einreichungen durch 10 Firmen
  - **Diagram Interchange:** 6 LOIs;
    3 Einreichungen durch 6 Firmen

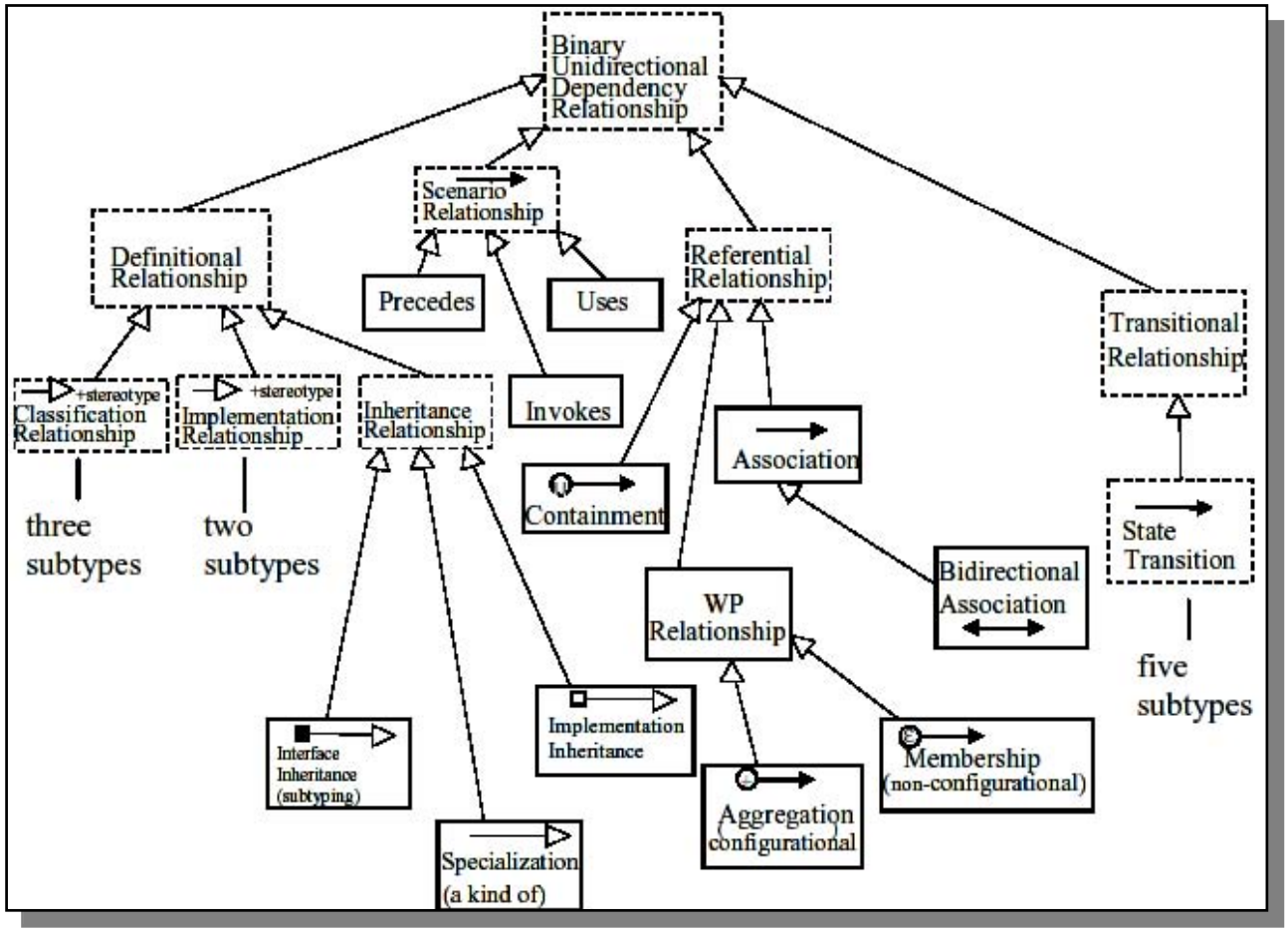
- Eingereicht durch Einzelfirmen und Konsortien

- Bezugnehmend auf einzelne Sprachaspekte der UML v1.x um diese neu zu erweitern; Vorschläge für vollständig neue Diagrammtypen oder die Abschaffung Existierender

Der Weg zu UML 2

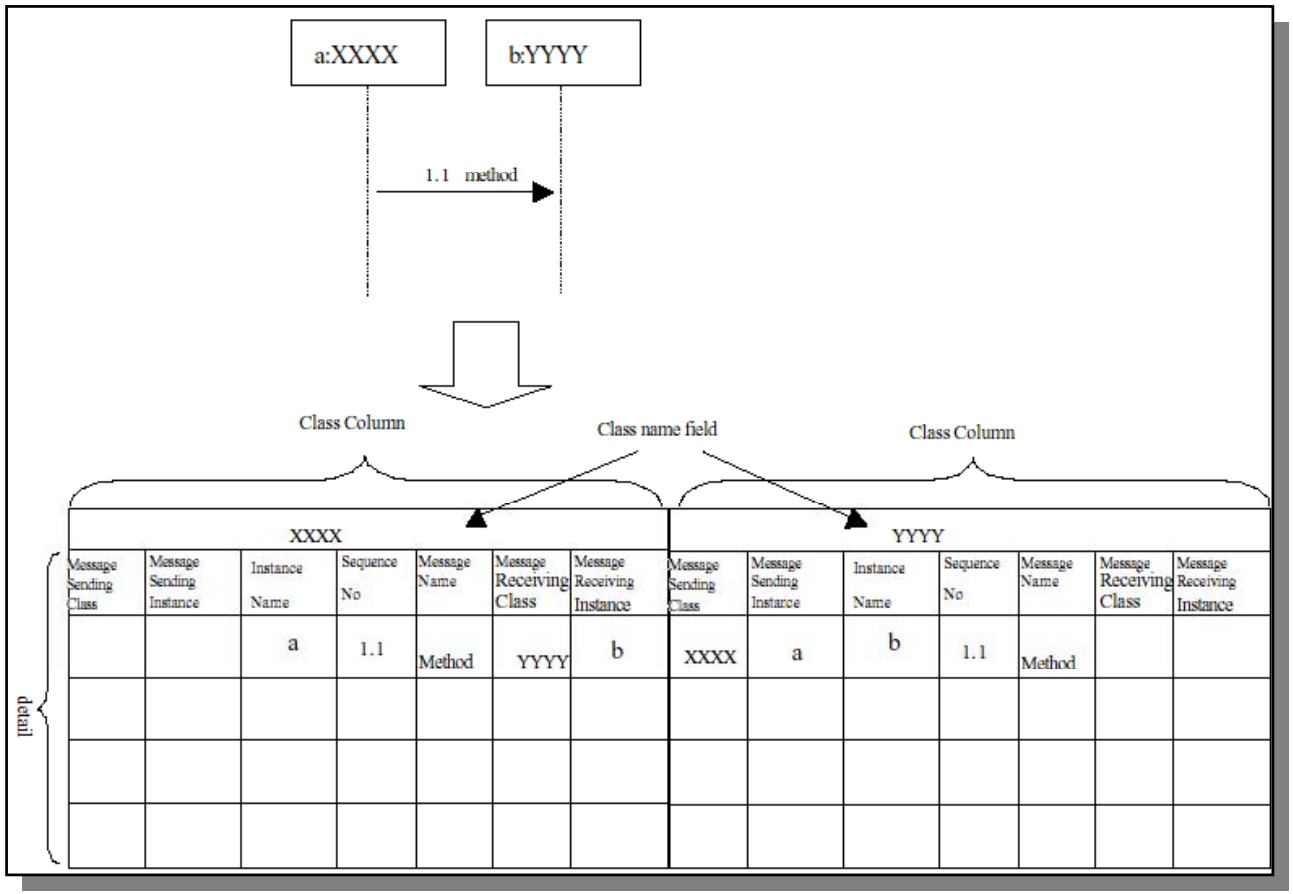


Vorschläge zur UML 2



Einige komplexe Dinge sollten einfacher werden ...

Vorschläge zur UML 2



Manchmal sagen Bilder einfach zu wenig ...

Vorschläge zur UML 2

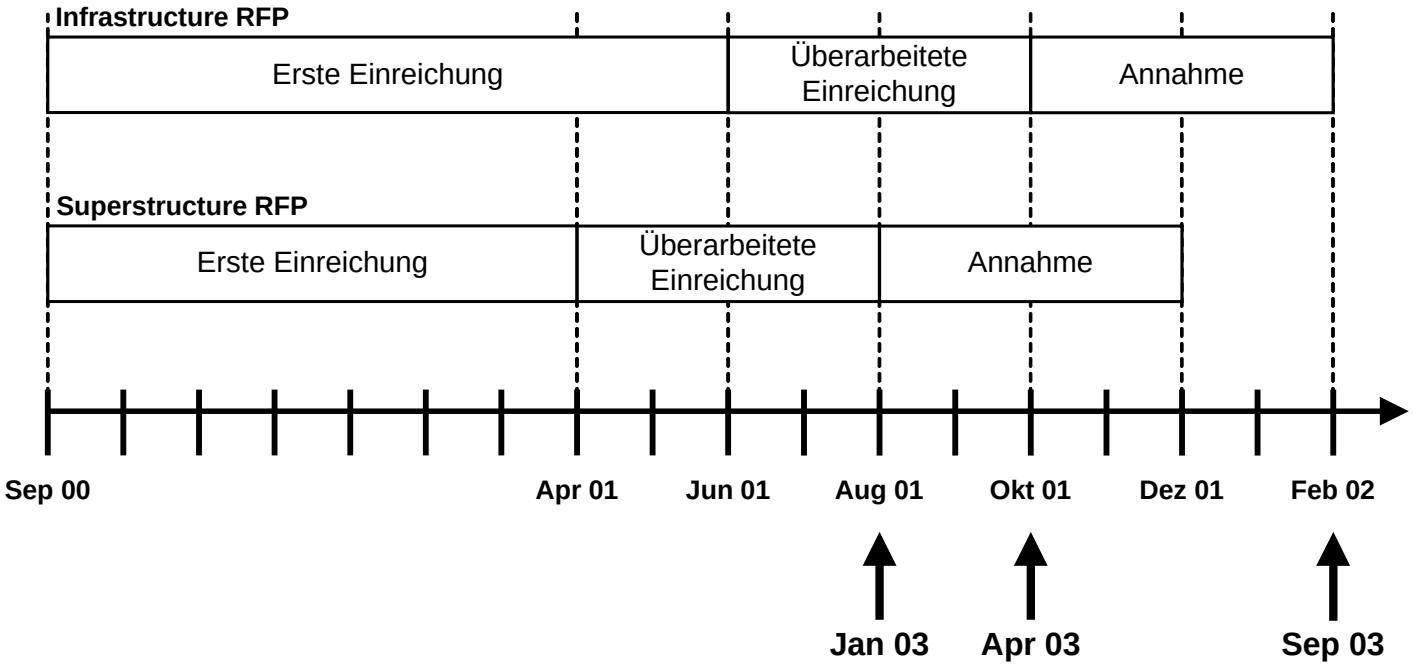
- ***Superstructure* und *Infrastructure:***
  Ausgereiftester und mit breiter Unterstützung bedachter Vorschlag durch die sog. „UML2 Partners“:

  - Mitglieder:
    Alcatel, Computer Associates, Ericsson, Hewlett-Packard, IONA, Kabira Technologies, Motorola, Oracle, Rational Software, SOFTEAM, Telelogic, and Unisys

  - Unterstützer:
    Advanced Concepts Center, Ceira Technologies, Compuware, Commisariat à L´Energie Atomique, DaimlerChrysler, Embarcardero Technologies, Enea Business Software, France Telecom, ...

UML 2.0 Ablaufplan

> Ziel: Ein UML 2.0 Standard



UML 2.0 Standardisierungsablauf

- Komplexer Annahmeprozess

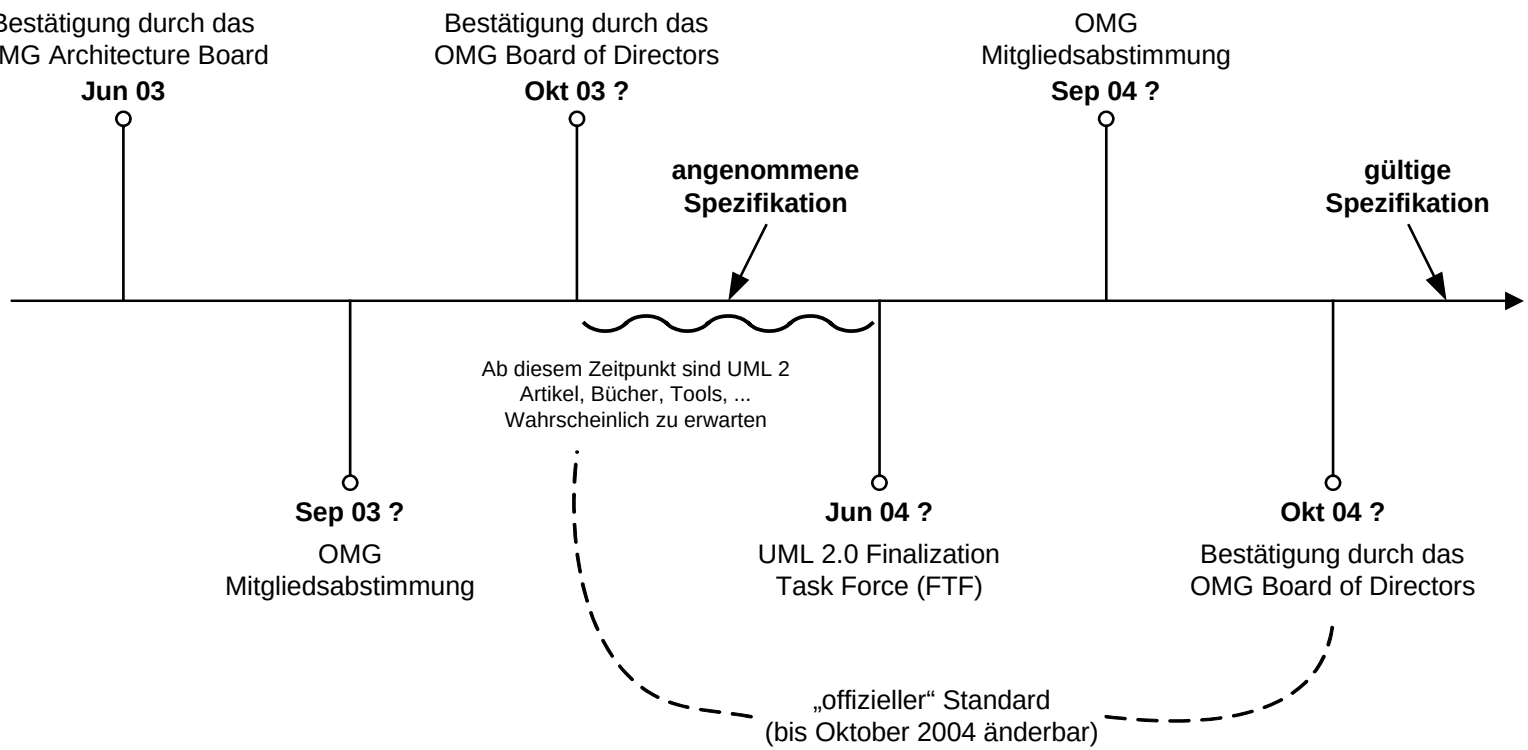
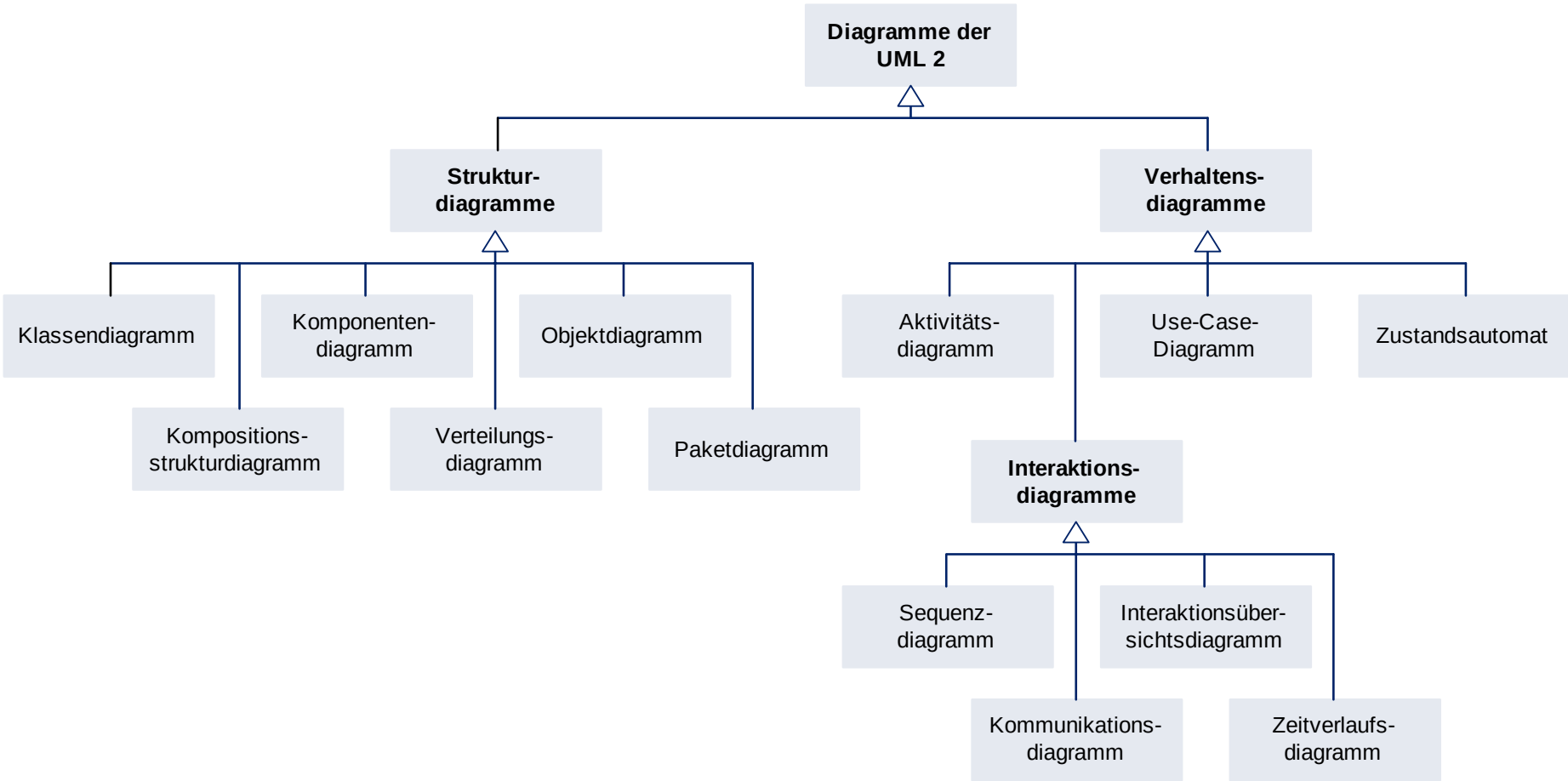


Diagramme der UML 2



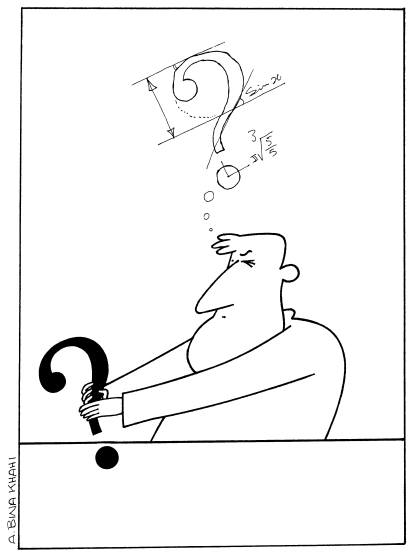
Wie hätten Sie´s denn gern?

Rational Unified Process

Arte

Agiles Vorgehen

Crystal



XP

ASD

Lean Software Development

Spiralmodell

V-Modell

Unified Process

SCRUM

Entscheidungshilfe (1): Gibt es Vorschriften?

Es gibt Vorschriften

Was wird durch Ihre Umwelt erzwungen?

- Standardkonformität erforderlich zwecks Freigabe des Systems (TÜV it, FDA,...)?
- Standard/Vorgehensmodell schreibt gewisse Notation vor
- Zertifizierbarkeit des Systems erforderlich?

Sie haben keine Wahl!

Allenfalls ergänzende Informationen in anderen Notationen hinzufügen und auf Vorschriften abbilden!

Es haben freie Wahl

Machen Sie das Beste daraus!

Entscheidunghilfe (2): Akzeptanz von Formalität

Leser akzeptiert Formalismus

Was wünschen/ akzeptieren die Stakeholder?

- Wie hoch ist die Motivation, sich in Dokumente des Entwicklungsprozesses zu vertiefen?
- Welche Grundausbildung haben die Betroffenen?
- Aufgeschlossenheit gegenüber neuen Notationen vorhanden?
- Starke Fixierung auf die alt bekannte Methode?

Leser akzeptiert keinerlei Formalismus

Prototypen jeglicher Art (Lo-fi, Hi-fi)

Entscheiden Sie zwischen „Lesbarkeit“, „Verständlichkeit“ auf der einen Seite und „Prüfbarkeit“, „Automatisier-barkeit“ auf der anderen Seite.

Entscheidungshilfe (3): Eignung für die Problemstellung

- .... Jetzt kommen endlich die Fragen nach dem fachlichen Inhalt, den ein Modell darstellt

Diagramme der UML und ihre Anwendung I

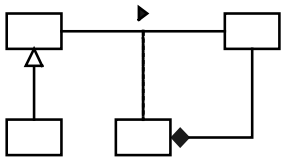
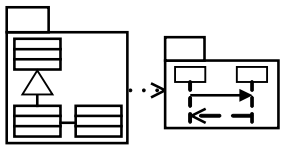
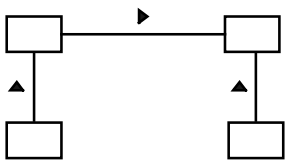
Diagrammtyp	Diese zentrale Frage beantwortet das Diagramm	Stärken
Klassendiagramm 	Aus welchen Klassen besteht mein System und wie stehen diese untereinander in Beziehung?	Beschreibt die statische Struktur des Systems. Enthält alle relevanten Struktur-zusammenhänge/Datentypen. Brücke zu dynamischen Diagrammen. Normalerweise unverzichtbar.
Paketdiagramm 	Wie kann ich mein Modell so schneiden, dass ich den Überblick bewahre?	Logische Zusammenfassung von Modellelementen. Modellierung von Abhängigkeiten/ Inklusion möglich.
Objektdiagramm 	Welche innere Struktur besitzt mein System zu einem bestimmten Zeitpunkt zur Laufzeit (Klassendiagramm-schnappschuss)?	Zeigt Objekte u. Attributbelegungen zu einem bestimmten Zeitpunkt. Verwendung beispielhaft zur Veranschaulichung Detailniveau wie im Klassen-diagramm. Sehr gute Darstellung von Mengenverhältnissen.

Diagramme der UML und ihre Anwendung II

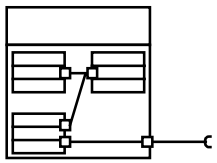
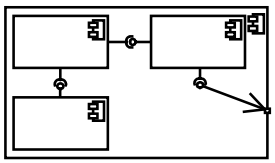
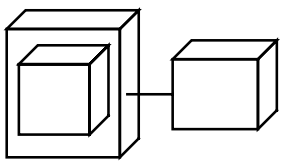
Diagrammtyp	Diese zentrale Frage beantwortet das Diagramm	Stärken
Kompositionsstruktur-diagramm 	Wie sieht das Innenleben einer Klasse, einer Komponente, eines Systemteils aus?	Ideal für die Top-Down-Modellierung des Systems (Ganz-Teil-Hierarchien). Zeigt Teile eines „Gesamtelements“ und deren Mengenverhältnisse. Präzise Modellierung der Teile-Beziehungen über spezielle Schnittstellen (Ports) möglich.
Komponentendiagramm 	Wie werden meine Klassen zu wieder verwendbaren, verwaltbaren Komponenten zusammengefasst und wie stehen diese in Beziehung?	Zeigt Organisation und Abhängig-keiten einzelner technischer Systemkomponenten. Modellierung angebotener und benötigter Schnittstellen möglich.
Verteilungsdiagramm 	Wie sieht das Einsatzumfeld (Hardware, Server, Datenbanken, …) des Systems aus? Wie werden die Komponenten zur Laufzeit wohin verteilt?	Zeigt das Laufzeitumfeld des Systems mit den „greifbaren“ Systemteilen. Darstellung von „Softwareservern“ möglich. Hohes Abstraktionsniveau, kaum Notationselemente.

Diagramme der UML und ihre Anwendung III

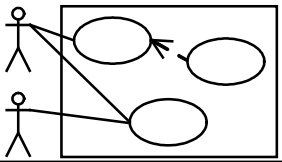
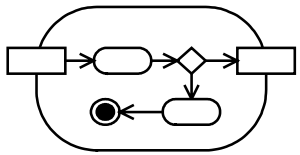
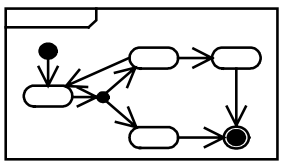
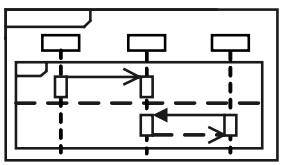
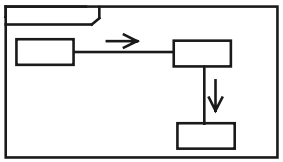
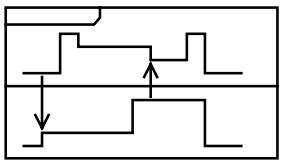
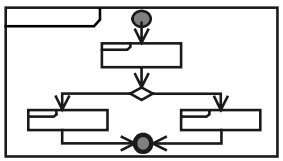
Diagrammtyp	Diese zentrale Frage beantwortet das Diagramm	Stärken
Use-Case-Diagramm 	Was leistet mein System für seine Umwelt (Nachbarsysteme, Stakeholder)?	Außensicht auf das System. Geeignet zur Kontextabgrenzung. Hohes Abstraktionsniveau, einfache Notationsmittel.
Aktivitätsdiagramm 	Wie läuft ein bestimmter fluss-orientierter Prozess oder ein Algorithmus ab?	Sehr detaillierte Visualisierung von Abläufen mit Bedingungen, Schleifen, Verzweigungen. Parallelisierung und Synchronisation. Darstellung von Datenflüssen.
Zustandsautomat 	Welche Zustände kann ein Objekt, eine Schnittstelle, ein Use Case, … bei welchen Ereignissen annehmen?	Präzise Abbildung eines Zustands-modells mit Zuständen, Ereignissen, Nebenläufigkeiten, Bedingungen, Ein- und Austrittsaktionen. Schachtelung möglich.
	Wer tauscht mit wem welche Informationen in welcher Reihenfolge aus?	Darstellung d. Informationsaustauschs zwischen Kommunikationspartnern Sehr präzise Darstellung der zeitlichen Abfolge auch mit Nebenläufigkeiten.

Diagramme der UML und ihre Anwendung IV

Diagrammtyp	Diese zentrale Frage beantwortet das Diagramm	Stärken
Kommunikations-diagramm 	Wer kommuniziert mit wem? Wer „arbeitet“ im System zusammen?	Stellt den Informationsaustausch zwischen Kommunikationspartnern dar. Überblick steht im Vordergrund (Details und zeitliche Abfolge weniger wichtig).
Timingdiagramm 	Wann befinden sich verschiedene Interaktionspartner in welchem Zustand?	Visualisiert das exakte zeitliche Verhalten von Klassen,Schnittstellen,.. Geeignet für die Detailbetrachtungen, bei denen es wichtig ist, dass ein Ereignis zum richtigen Zeitpunkt eintritt.
Interaktionsübersichts-diagramm 	Wann läuft welche Interaktion ab?	Verbindet Interaktionsdiagramme (Sequenz-, Kommunikation- und Timingdiagramme) auf Top-Level-Ebene. Hohes Abstraktionsniveau.

Anforderungen eingehalten?

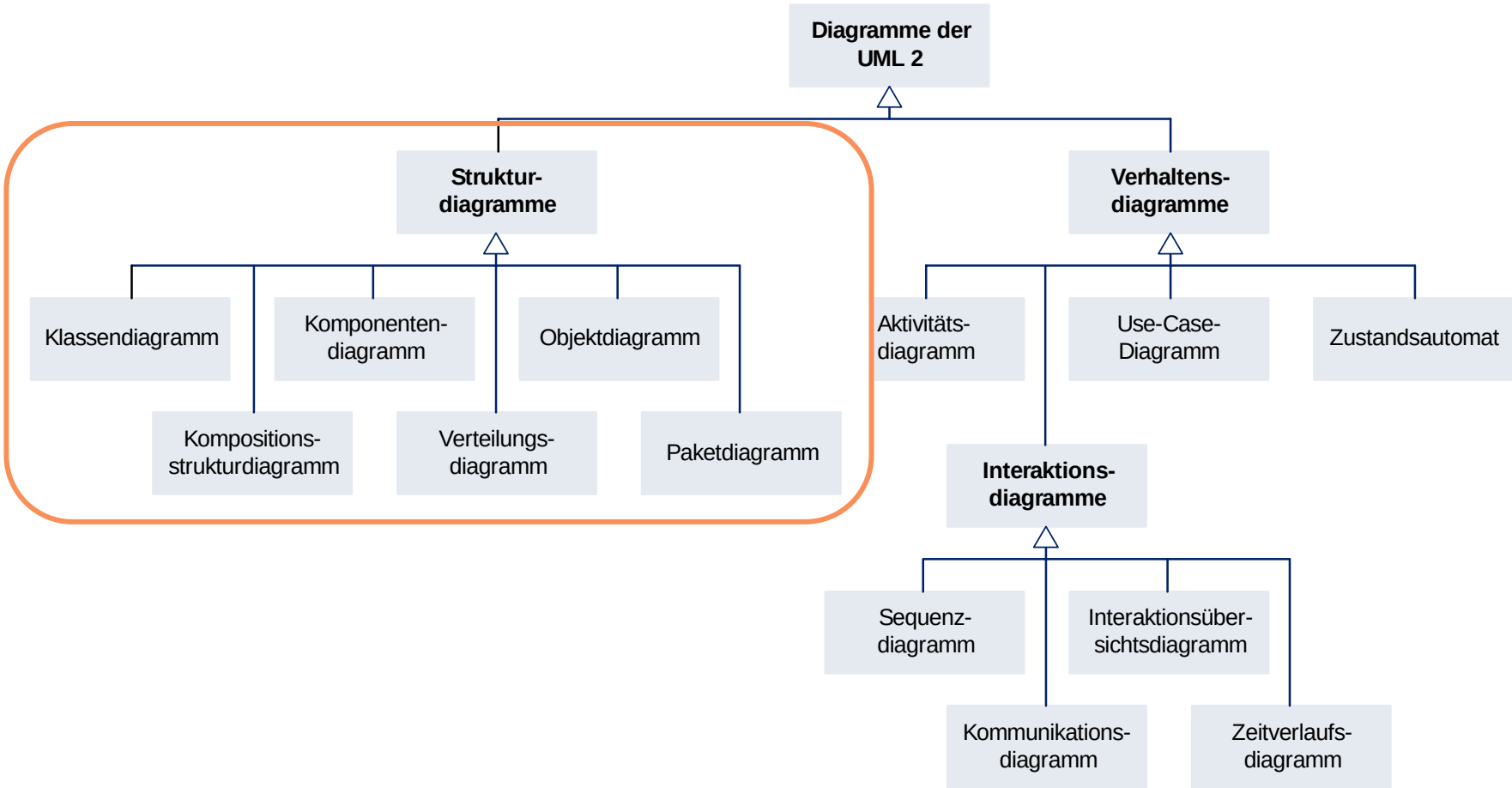
Die Bewertung

- UML kompakter und in sich schlüssiger durch Anpassung von Konzepten der Infrastruktur
- Angestrebte Abwärtskompatibilität zu älteren UML-Versionen in vielen Bereichen verletzt
- Die aktuelle Version der UML 2 Superstructure beinhaltet noch einige Kinderkrankheiten
- Das grafische Aussehen einiger Notationselemente hat sich verändert, was sicherlich zur Verwirrung führen wird
- Eingeführte Bezeichnungen und Namen wurden verändert

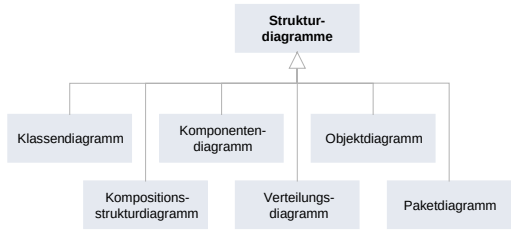
Umsteigen: Ja oder Nein?
Wann sich der Umstieg wirklich lohnt

- Die Modellierung des Systemverhaltens steht im Vordergrund: häufige Verwendung von Zustandsautomaten und Sequenz-diagrammen
- Modellierung im Umfeld technischer Systeme: bessere Darstellung z.B. der Kommunikation von Systemkomponenten oder des Zeitverhaltens eines Systems
- Modellierung von Geschäftsprozessen, bzw. von Aktivitäten eines Systems mittels Aktivitätsdiagrammen
- Generierung von Code oder Architekturen basierend auf einem MDA-Ansatz, oder in sich schnell ändernden, architektonischen Umfeldern wie EAI (Enterprise Application Integration)

Diagramme der UML 2



Die Strukturdiagramme

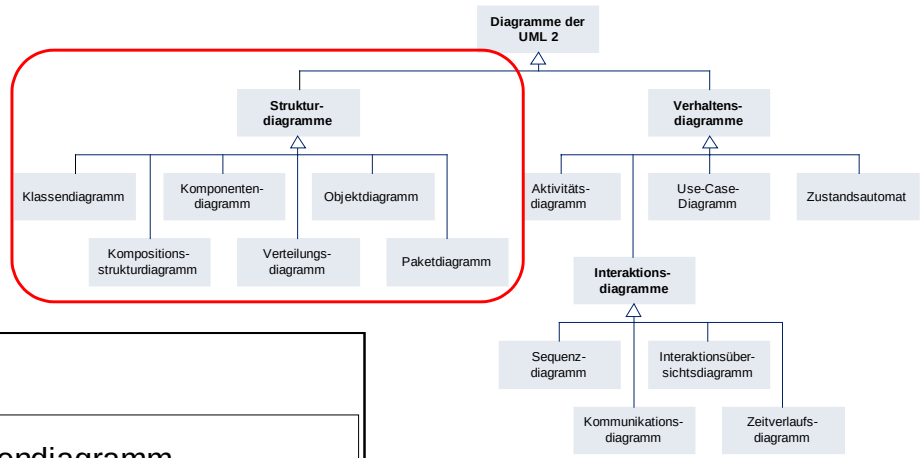


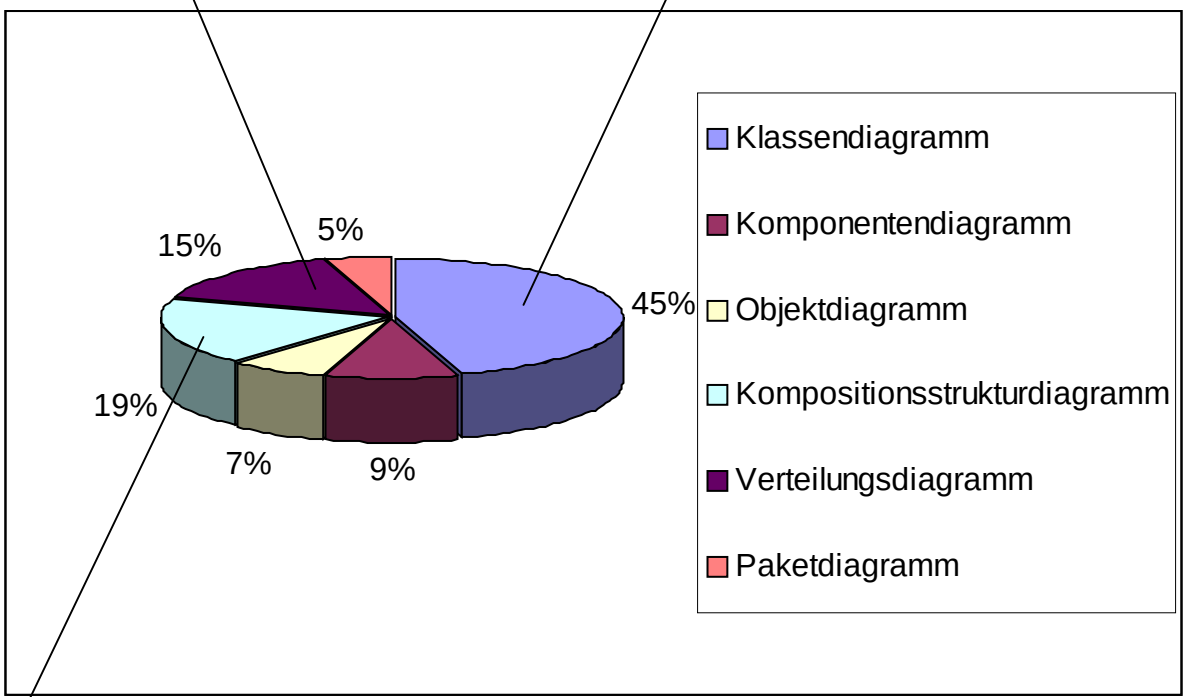
- **Klassendiagramm**
  Eine Zusammenstellung deklarativ-statischer Modellelemente (d.h. Klassen, Typen, ihre Inhalte und Beziehungen)
- **Objektdiagramm**
  Enthält Objekte und Beziehungsausprägungen
- **Paketdiagramm**
  Stellt die logische Organisation von Modellelementen und deren Abhängigkeiten dar
- **Komponentendiagramm**
  Zeigt die Organisation und Abhängigkeiten von Komponenten
- **Kompositionsstrukturdiagramm**
  Zeigt die interne Struktur eines *classifiers* sowie seine Möglichkeiten zu Interaktion mit anderen Systemkomponenten
- **Verteilungsdiagramm**
  Zeigt die Ausführungssicht des Systems

Statische Diagramme der UML 2

„wichtigster“ Diagrammtyp

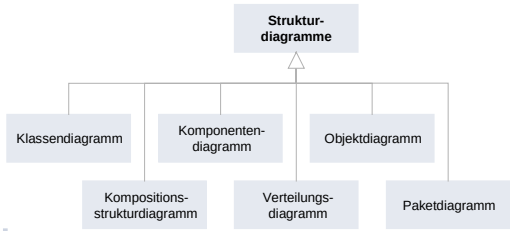
meist-geänderter Diagrammtyp



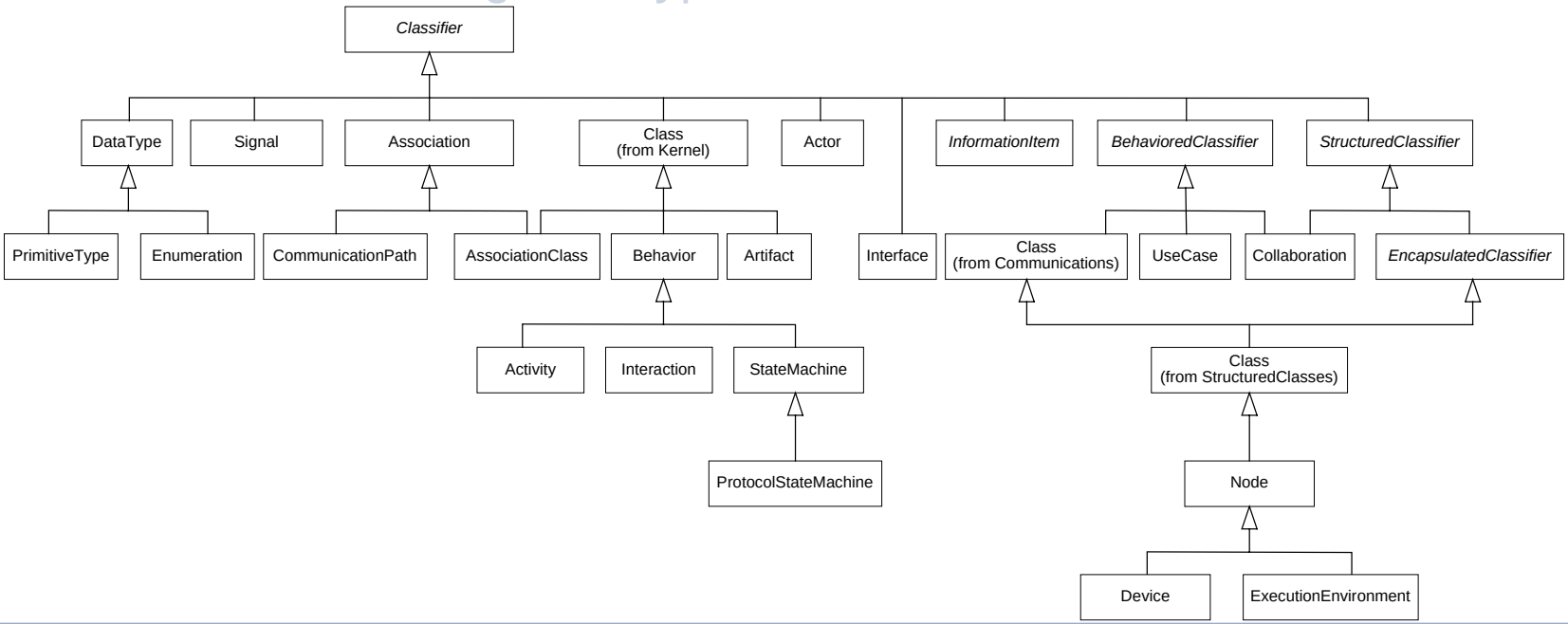


neuer Diagrammtyp

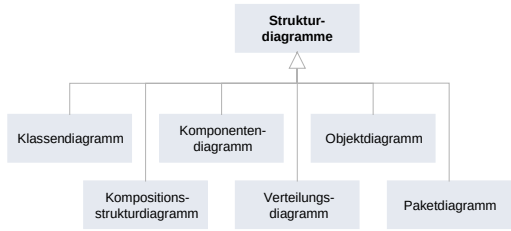
Basiskonzepte



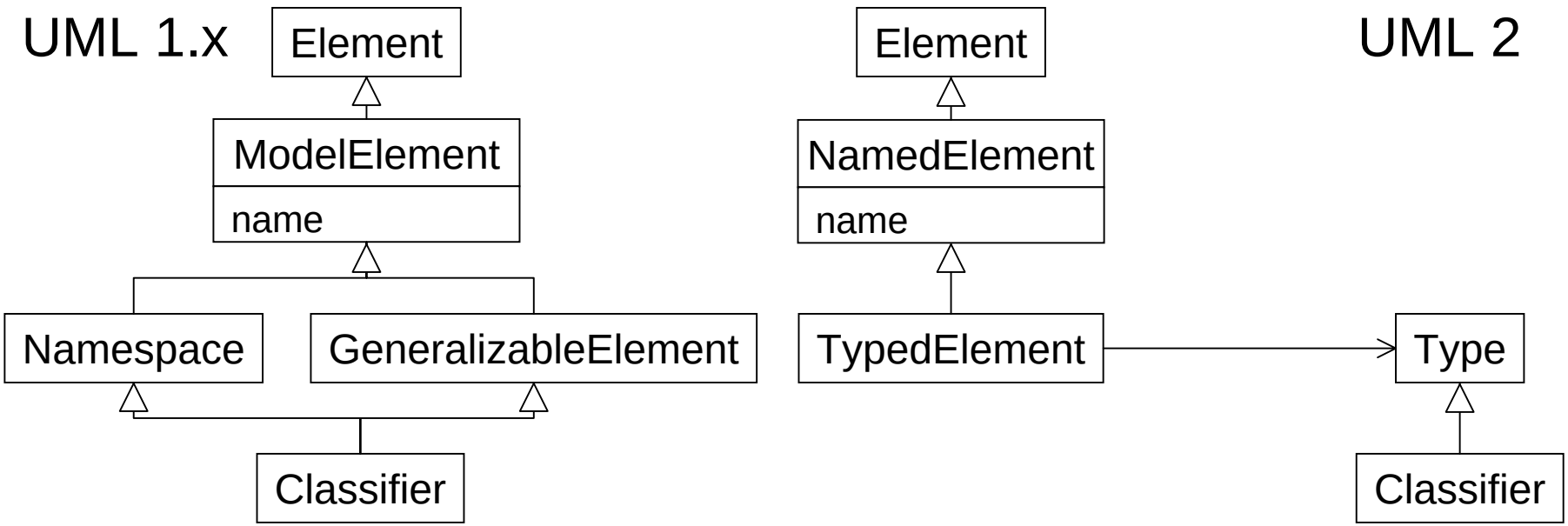
- UML 2 ...
  - Erweitert die Nutzung der *Classifier*
  - Vereinheitlicht ähnliche Konzepte unter gemeinsamen Oberbegriffen
  - Bietet neue Diagrammtypen und -sichten



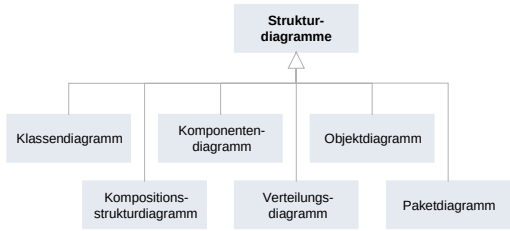
Basiskonzepte – *Classifier*



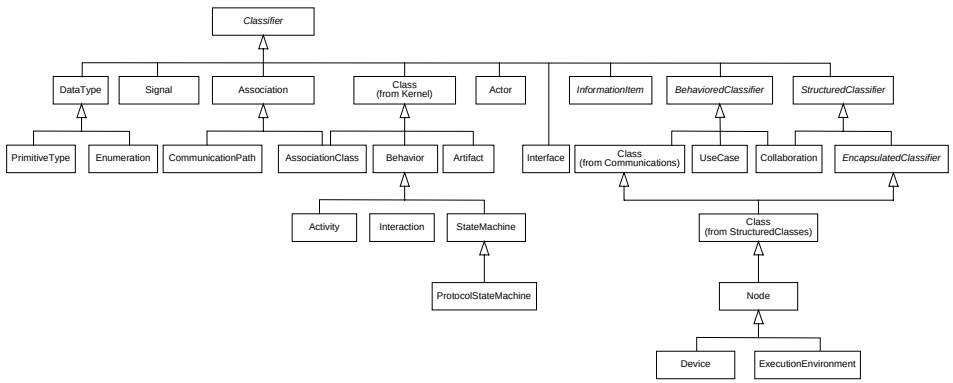
- UML 2 ...
  - Erweitert die Nutzung der *Classifier*
    - Verhältnis zwischen *Classifier* und *typisierten Elementen* klarer gefasst
    - Unterschied zwischen „typisiert sein“ und „Typ sein“ verwirklich



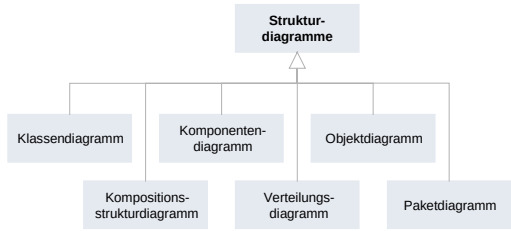
Basiskonzepte -- *Classifier*



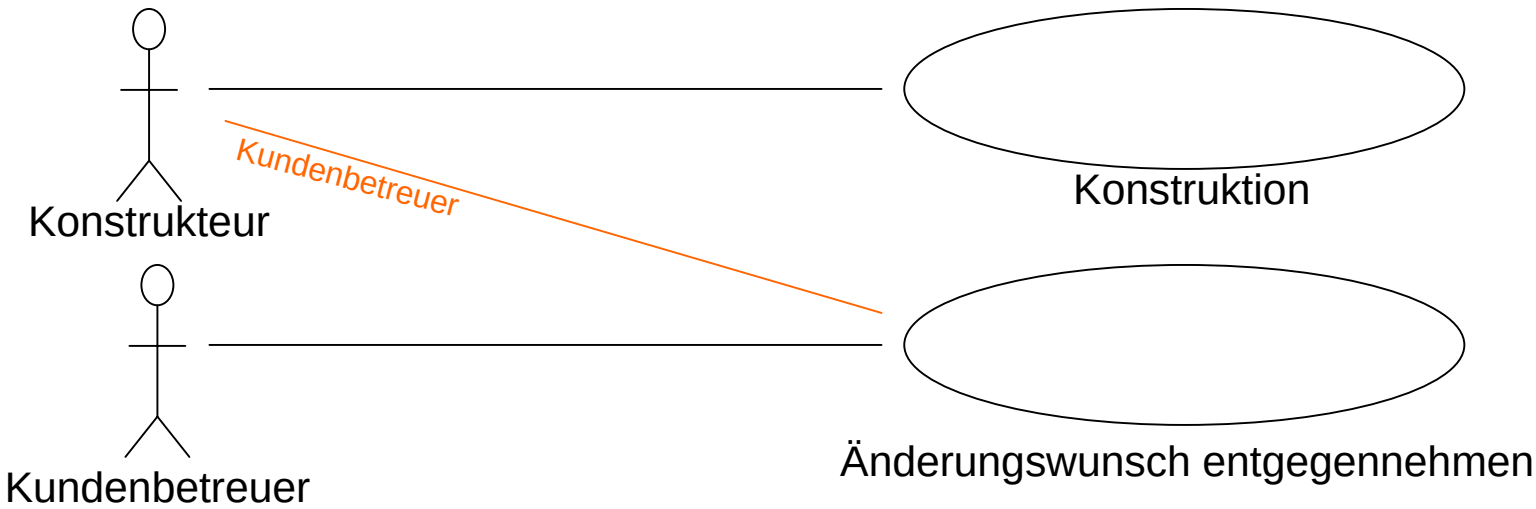
- UML 2 ...
  - Erweitert die Nutzung der *Classifier*
    - Beziehungen (*Assoziationen*) werden zwischen ihnen geknüpft
    - ... können Charakteristika (*Attribute*) besitzen
    - ... können Verhaltensspezifikationen (*Operationen*) besitzen
    - ... können generalisiert werden
    - ... können autonom auf Signale reagieren
    - ... können ausschließlich der Strukturierung dienen (*abstract*)



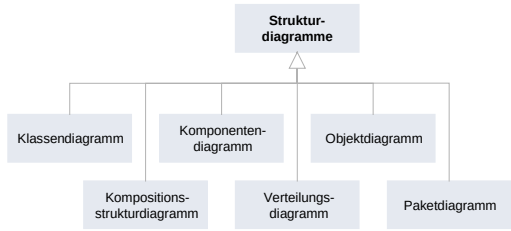
Basiskonzepte – Typisierung und Identität



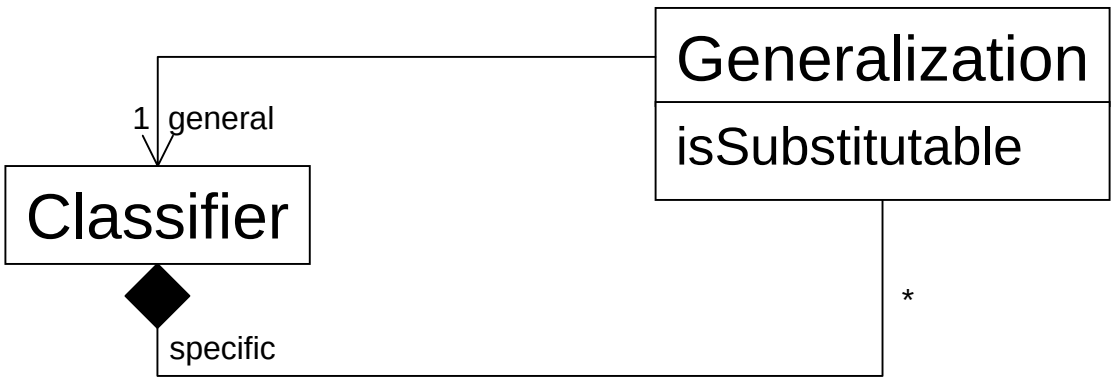
- UML 2 ...
  - Erweitert die Nutzung der *Classifier*
    - Beispiel: Assoziationen im Use-Case Diagramm



Basiskonzepte – Typisierung und Identität

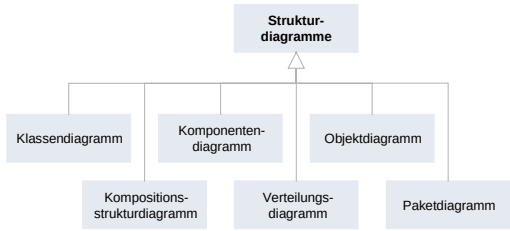


- UML 2 ...
  - Erweitert die Nutzung der *Classifier*
    - Beispiel: Generalisierung



- Alle Classifier-Spezialisierungen sind nun spezialisierbar
- Substituierbarkeit expliziert

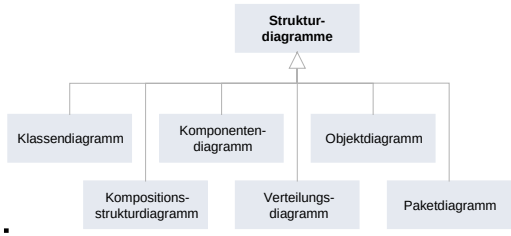
Basiskonzepte – Typisierung und Identität



- UML 2 ...
  - Erweitert die Nutzung der *Classifier*
    - Identitätsverhalten geklärt (Insbesondere bei dynamischer Verarbeitung statischer Daten)

- UML 2 ...
  - Identitätsbesitz ist nicht mehr statisch über die gesamte Lebensdauer
  - Identitätsbehaftete Objekte können durch dynamische Aktivitäten ihre Identität verlieren oder wechseln
  - Teilweise (beispielsweise in Kollaborationen) ist sie sogar unerheblich

Basiskonzepte

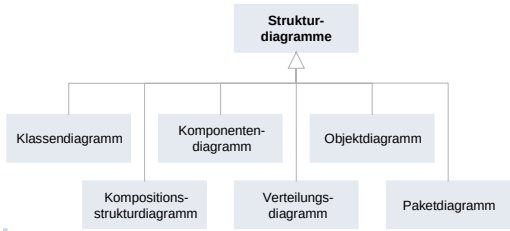


- UML 2 ...
  - Erweitert die Nutzung der Classifier
  - Vereinheitlicht ähnliche Konzepte unter gemeinsamen Oberbegriffen
  - Bietet neue Diagrammtypen und -sichten

- Beispiel:
  - *Artefakt* kann beliebige paktierbare Elemente „manifestieren“

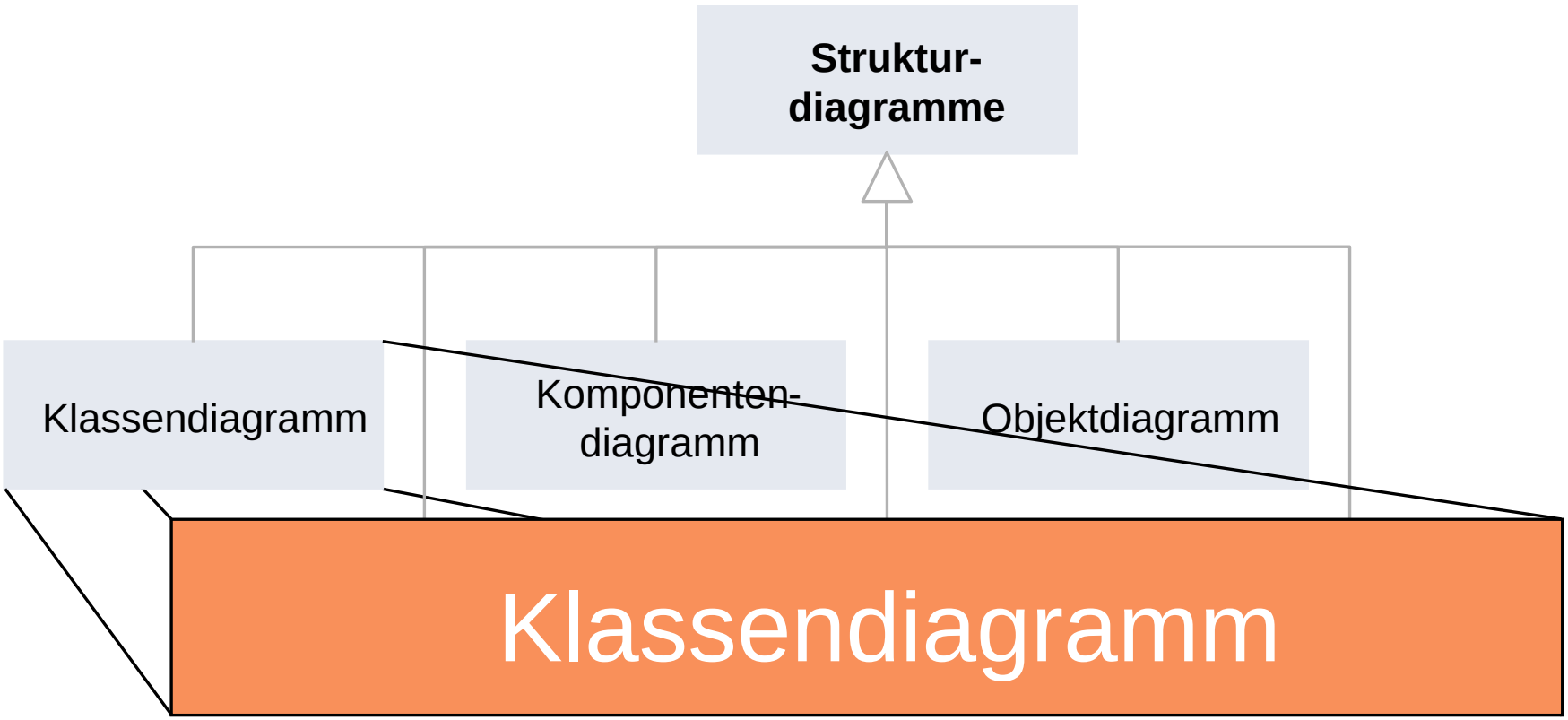
- Anwendung:
  - *Deployment* von Webapplikationen

Basiskonzepte

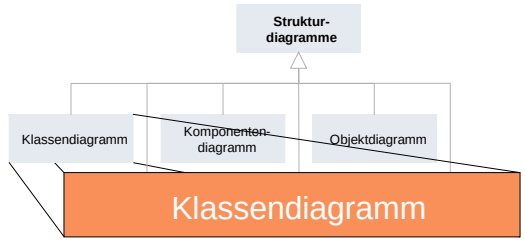


- UML 2 ...
  - Erweitert die Nutzung der Classifier
  - Vereinheitlicht ähnliche Konzepte unter gemeinsamen Oberbegriffen
  - Bietet neue Diagrammtypen und -sichten
- Beispiel:
  - Kompositionsstrukturdiagramm
    Zeigt die interne Struktur eines Classifiers sowie seine Möglichkeiten zu Interaktion mit anderen Systemkomponenten

Klassendiagramm --- Das „wichtigste“ Diagramm



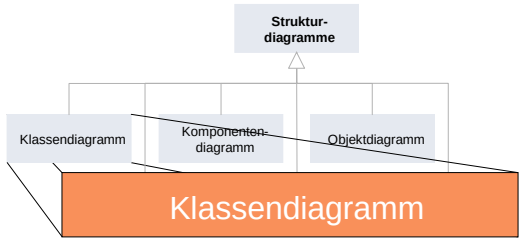
Klassendiagramm

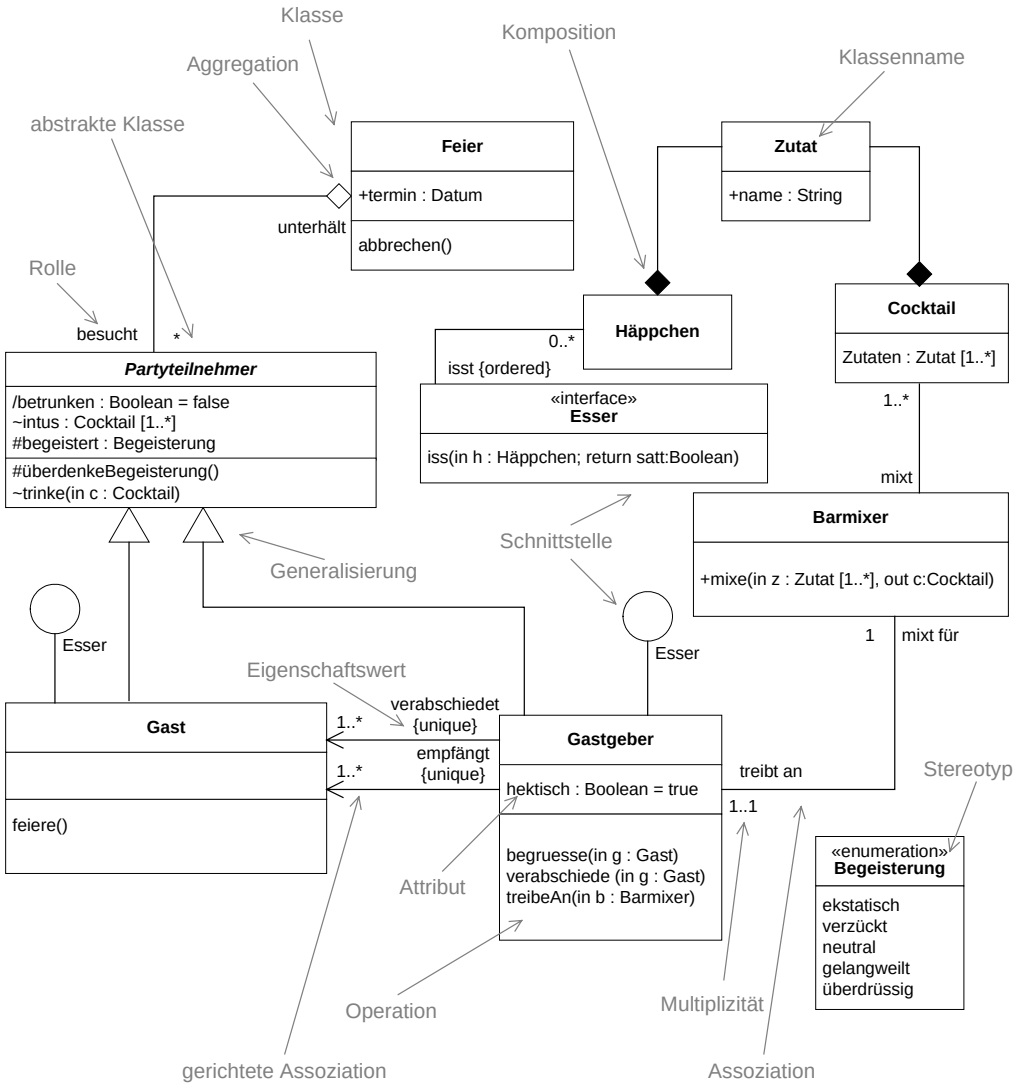


- **Funktion**:
  Zusammenstellung deklarativ-statischer Modellelemente (d.h. Klassen, Typen, ihre Inhalte und Beziehungen)

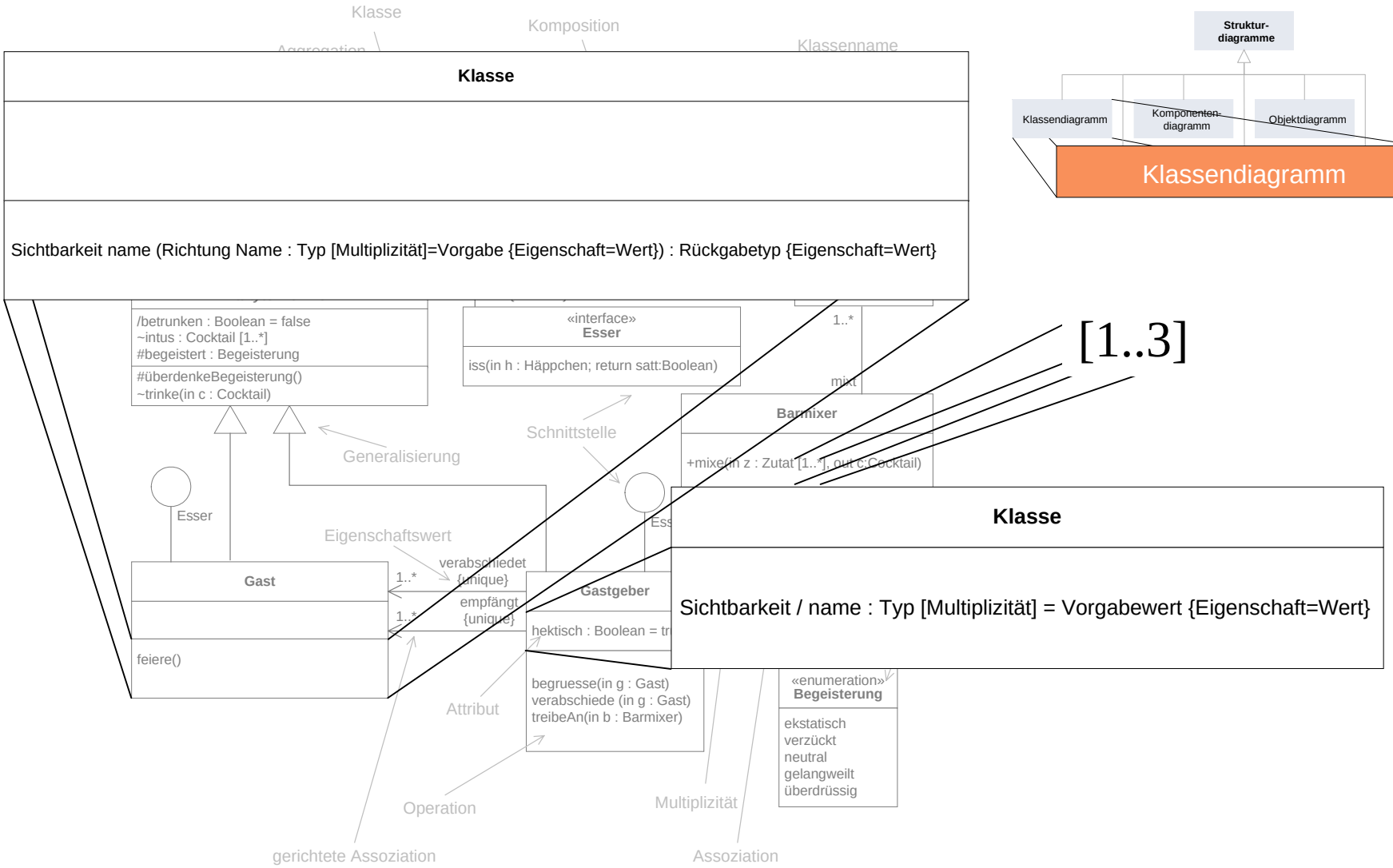
- **Aufgabe im Projekt**:
  - Variierend ...
    - von der ersten Darstellung konzeptueller Dateninhalte
    - über plattformunabhängige logische Modelle
    - bis hin zu „Implementierungsbauplänen“
      („Bilder-für-Java“, „Kästchen-und-Strichchen-statt-C++“)
  - Ersetzt oftmals das klassische, in Entity Relationship-Notation abgefasste, Datenmodell

Klassendiagramm

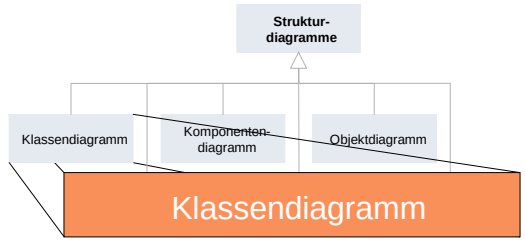




Klassendiagramm

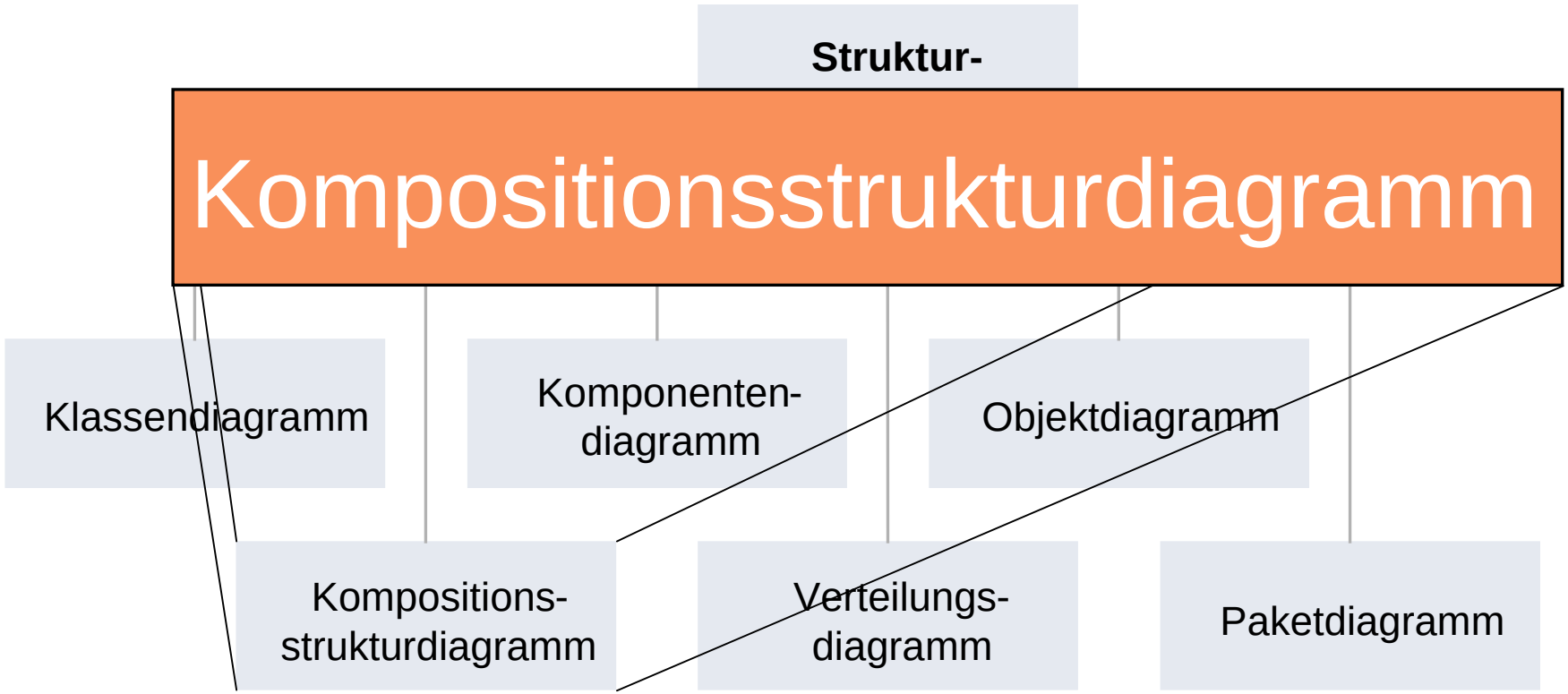


Klassendiagramm

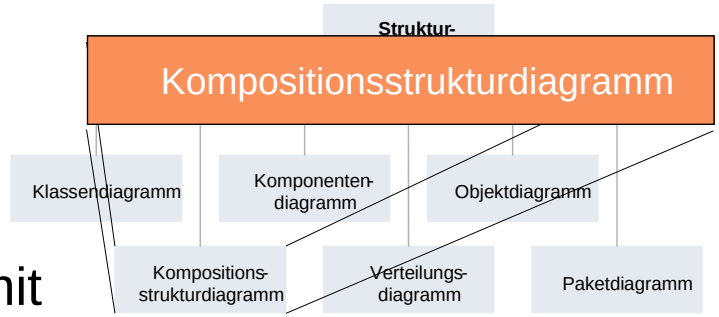


- Neu in UML 2:
  - Attribute besitzen Ordnung
  - Graphische Assoziationsnotation
    (Nutzung des „großen leeren Diamanten“auch für binäre Assoziationen)
  - Kommentarnotation „mit Punkt“
- Geändert in UML 2:
  - Graphische Schnittstellennotation (*Lolli pops*) jetzt Standard
  - Spezifikation von Sichtbarkeit, Name, Typ und Multiplizität für Operationen und Attribute vereinheitlicht
  - Multiplizitätsdefinition schärfer formuliert
  - Attribute sind keine implizite Kompositionsassoziation mehr
- Entfällt in UML 2:
  - Programmiersprachenmanifestationsspezifische Eigenschaften (z.B. *friend*)
  - Eigenschaften und Stereotypen unklarer Semantik
    (z.B. *become* und *copy*)

Kompositionsstrukturdiagramm --- Ein neues Diagramm in UML 2



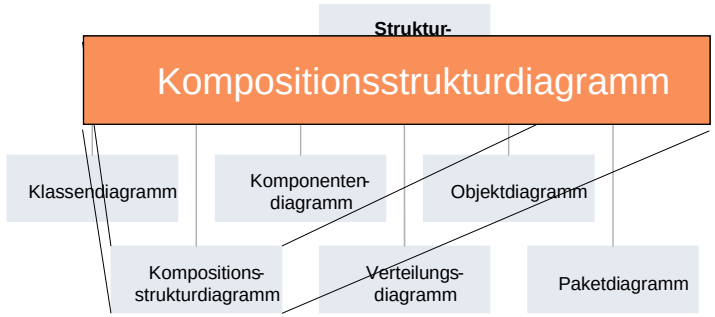
Kompositionsstrukturdiagramm



- **Funktion**:
  Zeigt die interne Struktur eines *Classifiers* sowie seine Möglichkeiten zu Interaktion mit anderen Systemkomponenten

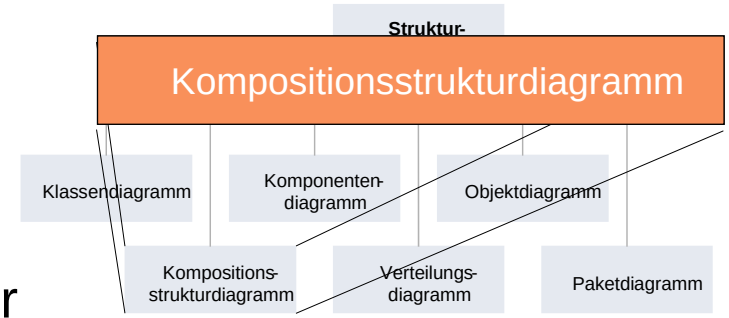
- **Aufgabe im Projekt**:
  - Beschreibung von extern angebotenen Schnittstellen
  - Abstraktion der Operationen und des Signals

Kompositionsstrukturdiagramm

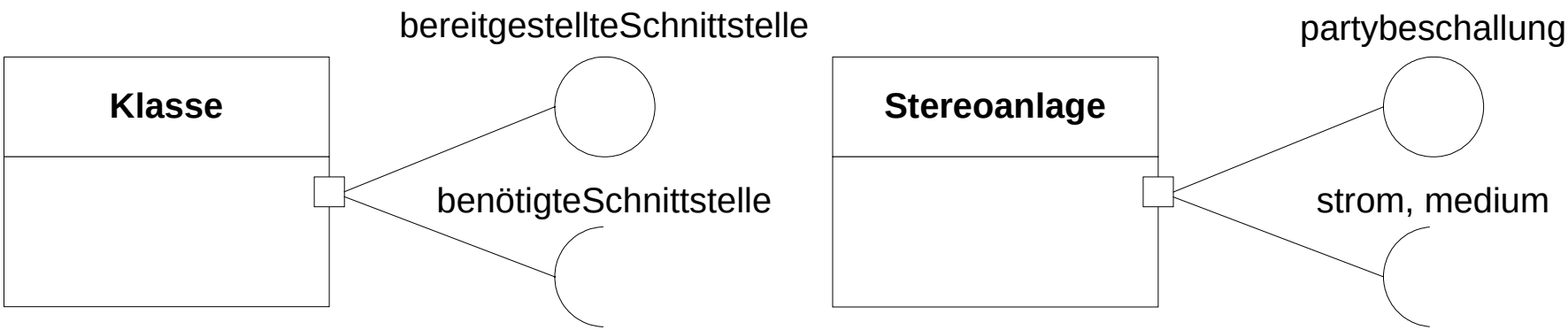


- Basiskonzepte:
  - **Part**
    Objekte oder Rollenausprägungen
  - **Port**
    Öffentlich sicht- und zugreifbarer Interaktionspunkt, der auf einen Operationsaufruf oder den Empfang eines Signals reagiert
  - **Kollaboration**
    Zusammenspiel von Operationen oder Classifiern
  - **Konnektor**
    Assoziationsinstanz (*Link*), die Kommunikation zwischen (allgemeinen) Instanzen ermöglicht
  - **Rollenverwendung**
    Bindet realisierende *Classifier* an Kollaborationsinstanz

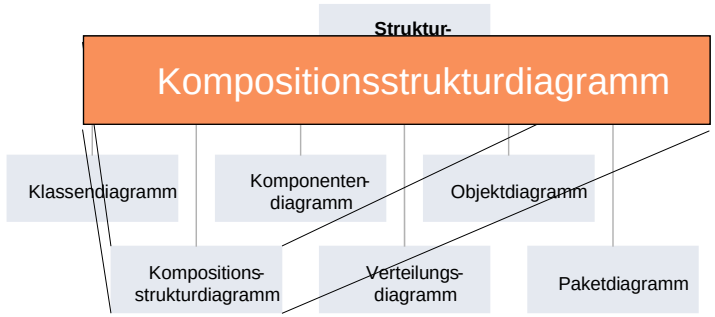
Kompositionsstrukturdiagramm



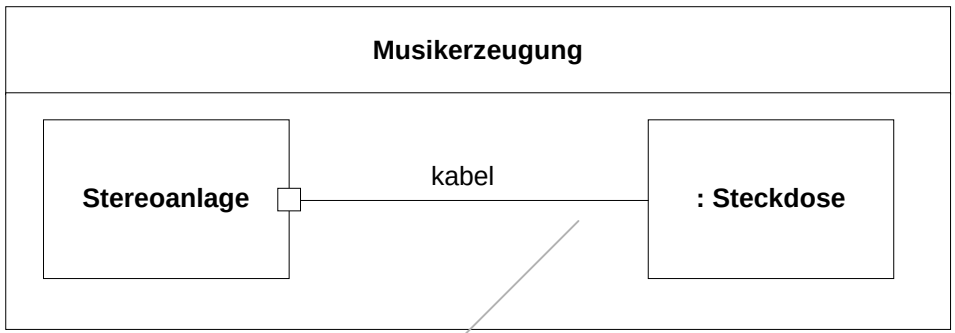
- Neu in UML 2:
  - **Port:** Definiert einen öffentlich sicht- und zugreifbaren Interaktionspunkt, der auf einen Operationsaufruf oder den Empfang eines Signals reagiert.



Kompositionsstrukturdiagramm

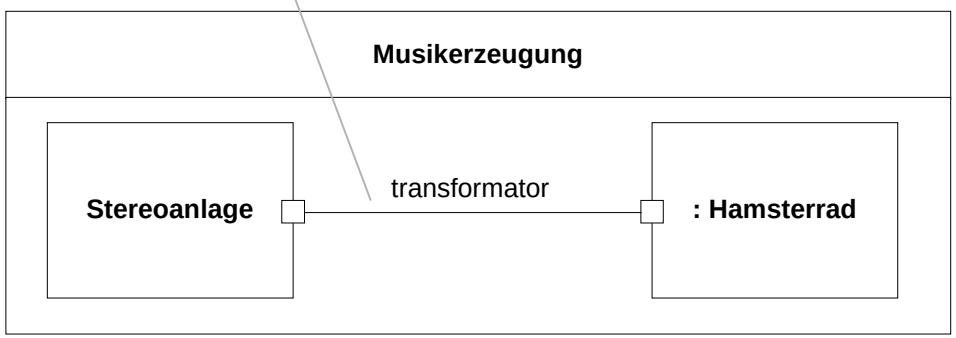


- Neu in UML 2:
  - **Port:** Werden durch andere *Classifier* oder Ports angesprochen.

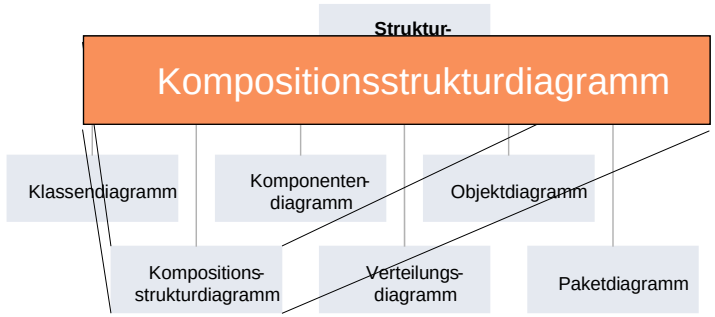


Konnektor

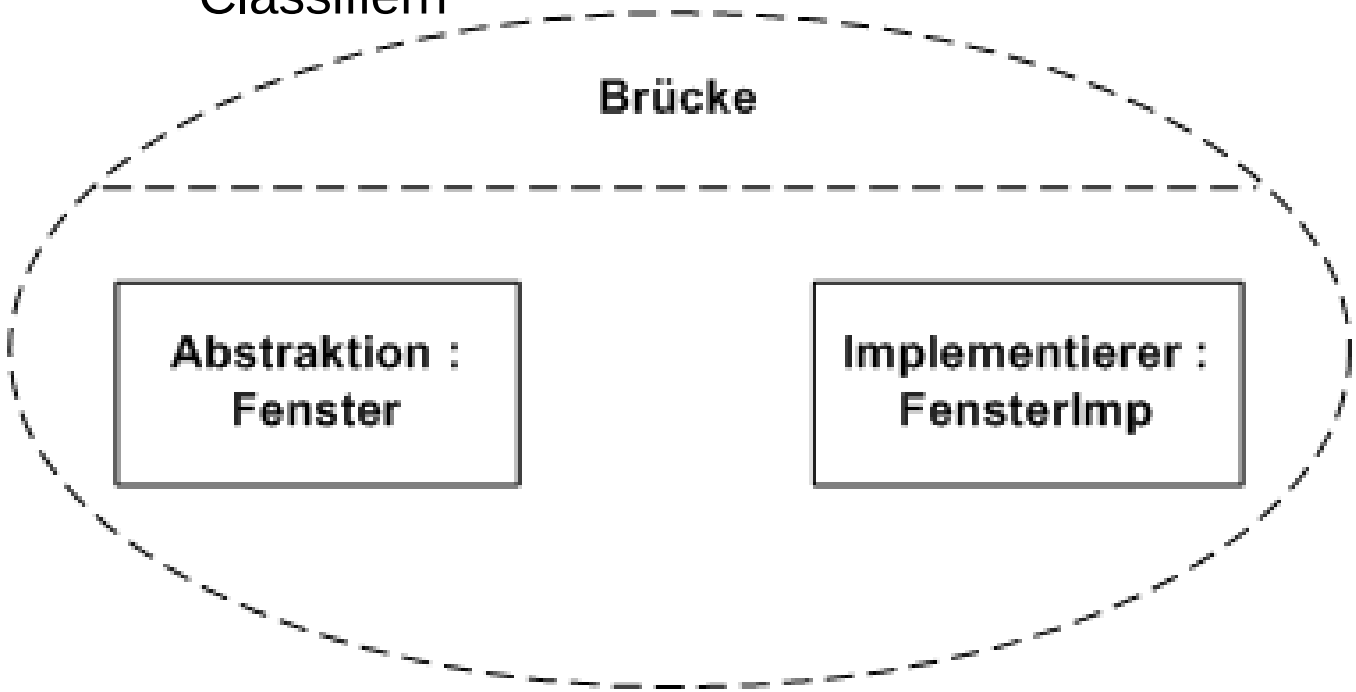
Konnektor



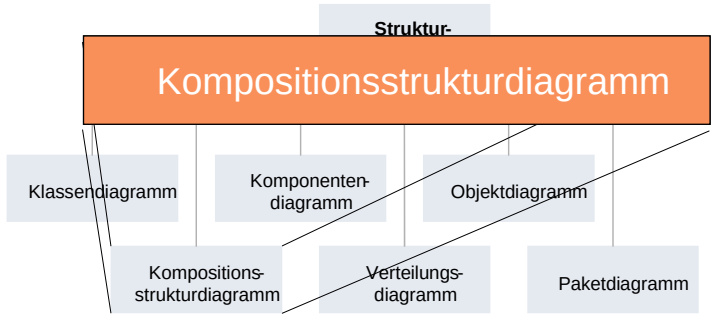
Kompositionsstrukturdiagramm



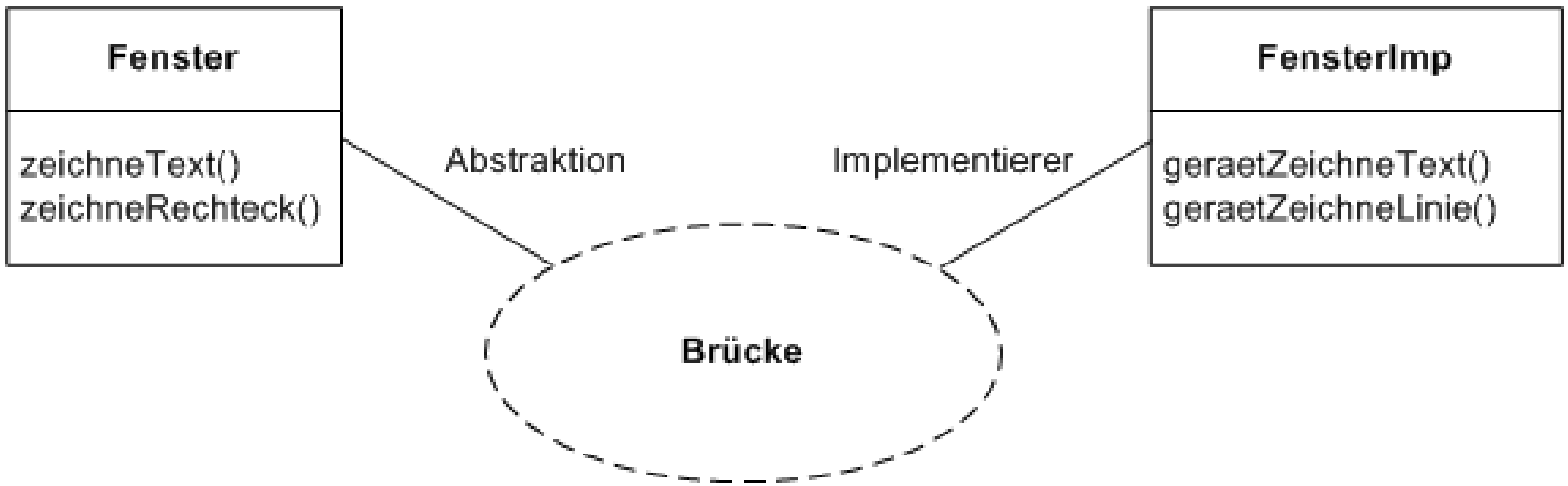
- Überarbeitet in UML 2:
  - **Kollaboration:** Visualisiert das Zusammenspiel von Operationen oder Classifiern



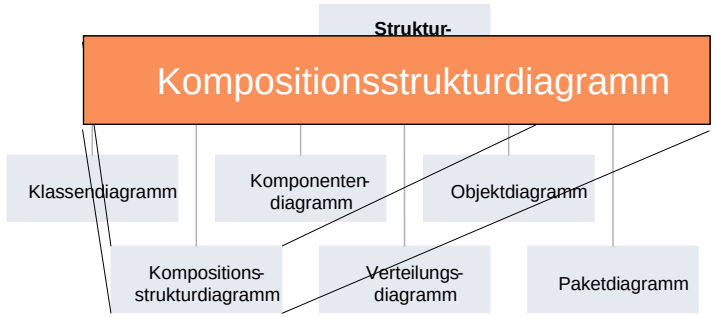
Kompositionsstrukturdiagramm



- Überarbeitet in UML 2:
  - **Kollaboration:** Visualisiert das Zusammenspiel von Operationen oder Classifiern

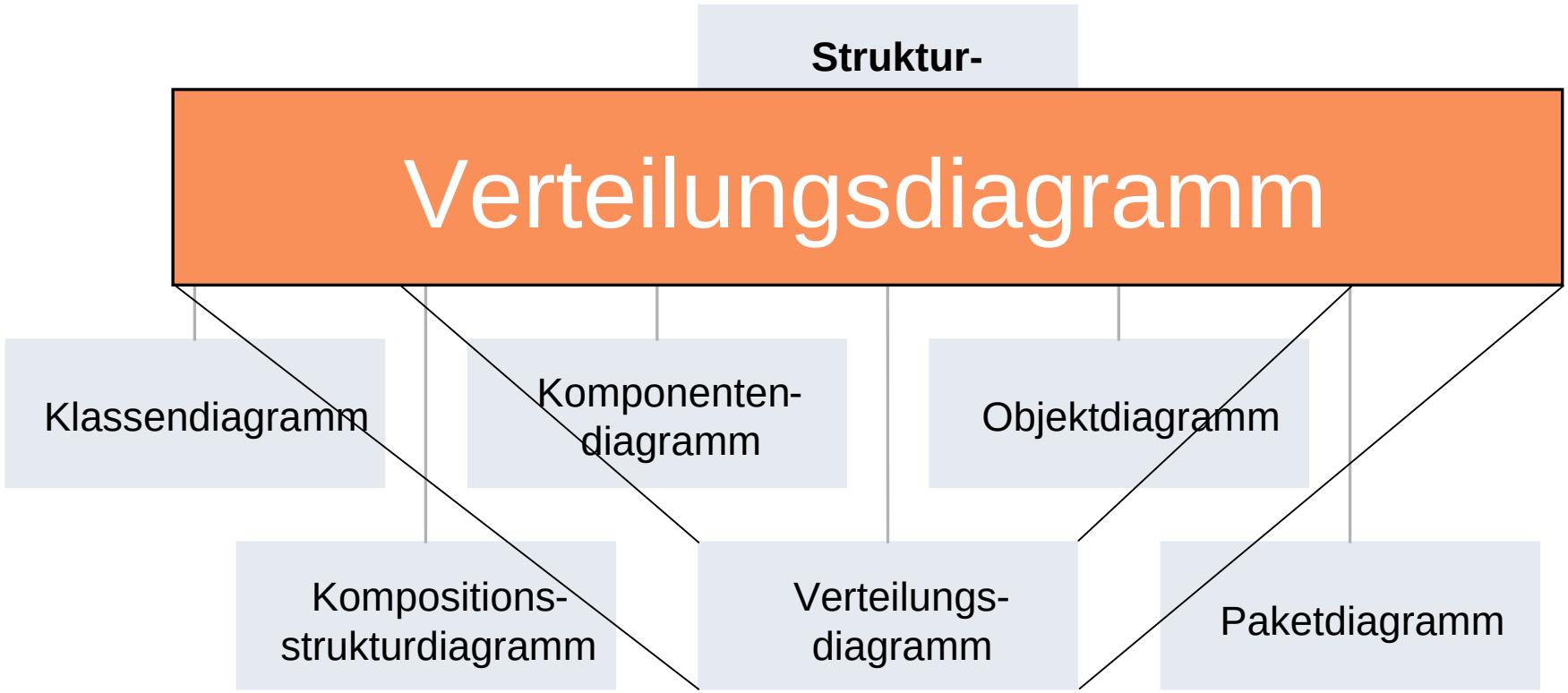


Kompositionsstrukturdiagramm

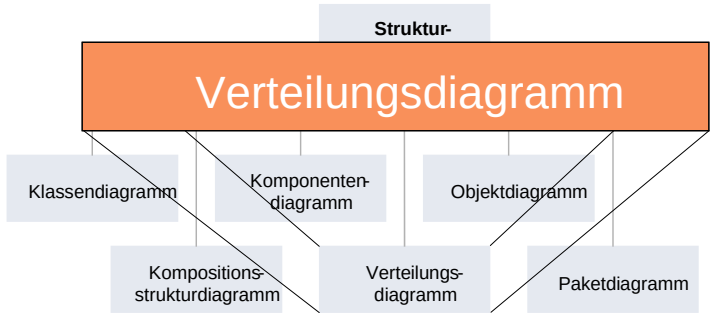


- Neu in UML 2:
  - Der gesamte Diagrammtyp
  - Ports als gemeinsame Abstraktion von Operationen und Signalen
  - Auffassung von Kollaborationen als Teile von Kompositionsstrukturen
- Geändert in UML 2:
  - Schnittstellennotation
  - Möglichkeiten zur Darstellung von Kollaborationen

Verteilungsdiagramm ---
Das Diagramm mit den meisten Änderungen



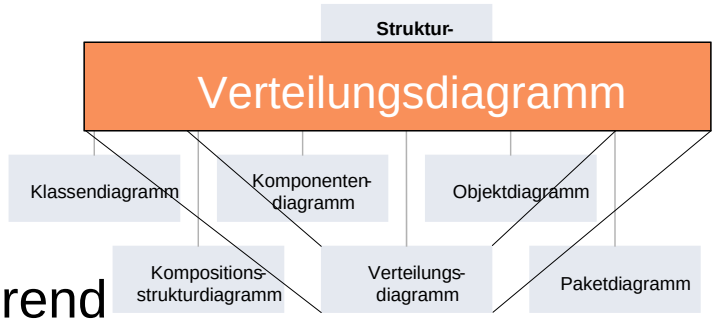
Verteilungsdiagramm



- **Funktion**:
  Zeigt die Ausführungssicht des Systems

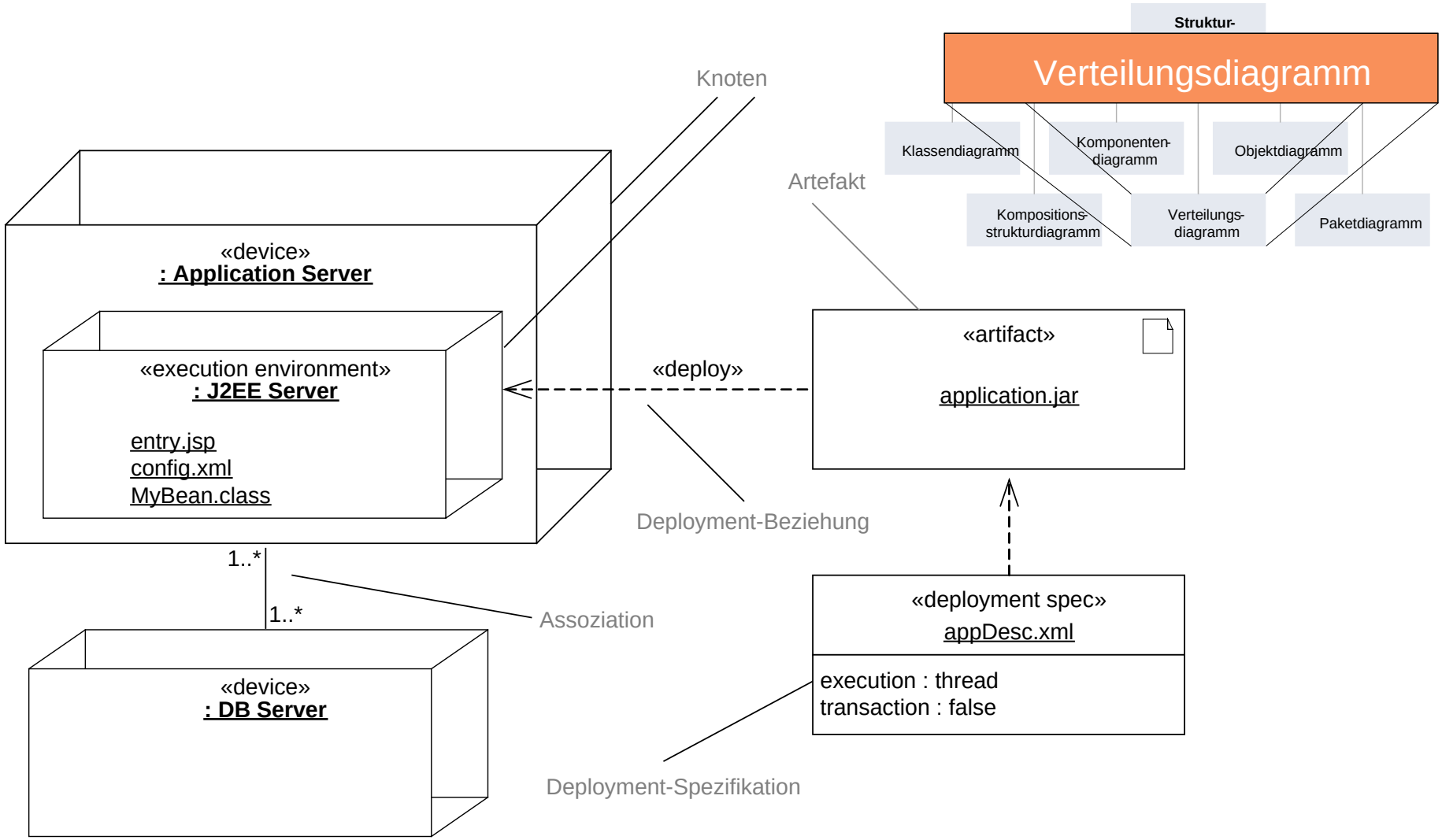
- **Aufgabe im Projekt**:
  - Beschreibung der Systemarchitektur
  - Erweitert Systemsicht um an der Ausführung beteiligte oder zur Ausführung benötigte Hardwarekomponenten

Verteilungsdiagramm

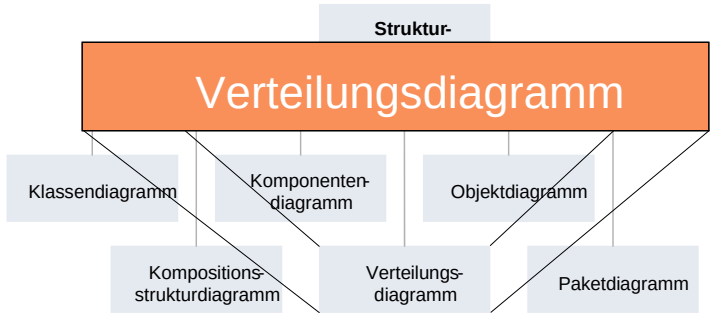


- Basiskonzepte:
  - **Artefakt**
    Physische Informationseinheit, die während des Entwicklungsprozesses erzeugt oder verwendet wird.
    (z.B. Modelle, Quellcode, Objektdateien, ...)
  - **Knoten**
    Classifier, der eine zur Ausführungszeit verfügbare Ressource darstellt.
  - **Einsatzspezifikation** (Deployment Specification)
    Parametermenge, die Laufzeitverhalten festlegt

Verteilungsdiagramm



Verteilungsdiagramm

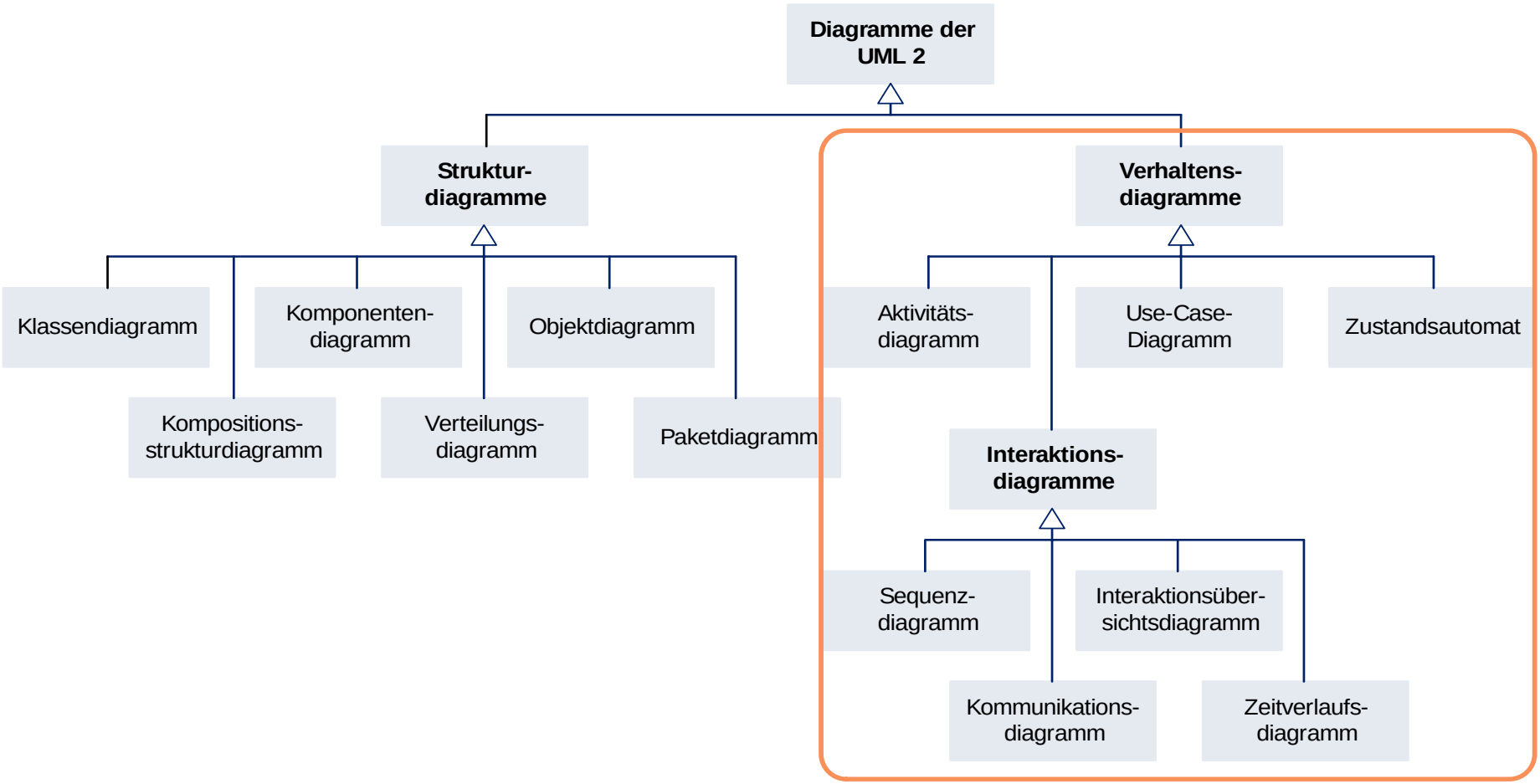


- Neu in UML 2:
  - Aktuelle Ausführungs- und Laufzeitumgebungen wie J2EE und .NET werden besser berücksichtigt
- Geändert in UML 2:
  - Knoten können hinsichtlich ihrer Rollen zum Ausführungszeitpunkt feinspezifiziert werden
  - Ausdetaillierung der Abhängigkeitsbeziehungen
- Entfällt in UML 2:
  - Unpraktikable graphische Darstellung des Knotens

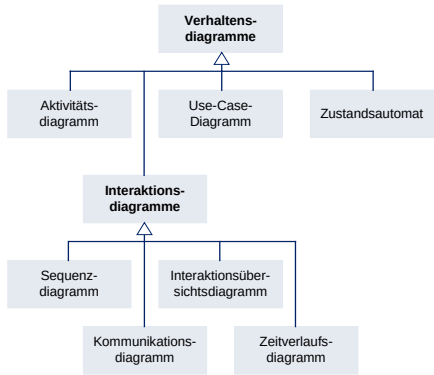
Fazit ---
statische Diagramme in UML 2

- Absichtsvoll wenig „spürbar“ Neues
  - Im Hintergrund
    - Klarere Basiskonzepte
    - Deutlich mehr Wiederverwendung
    - Fast vollständig neues Metamodell
- Modifizierte Semantik vieler (Meta-)Modellelemente
- Einige neue graphische Primitive
- Reichhaltigeres Angebot an Abhängigkeiten und „eingebauten“ Modellerweiterungen (Stereotypen)
- Behutsame Verrentung „problematischer“ Modellelemente

Diagramme der UML 2



Die Verhaltensdiagramme



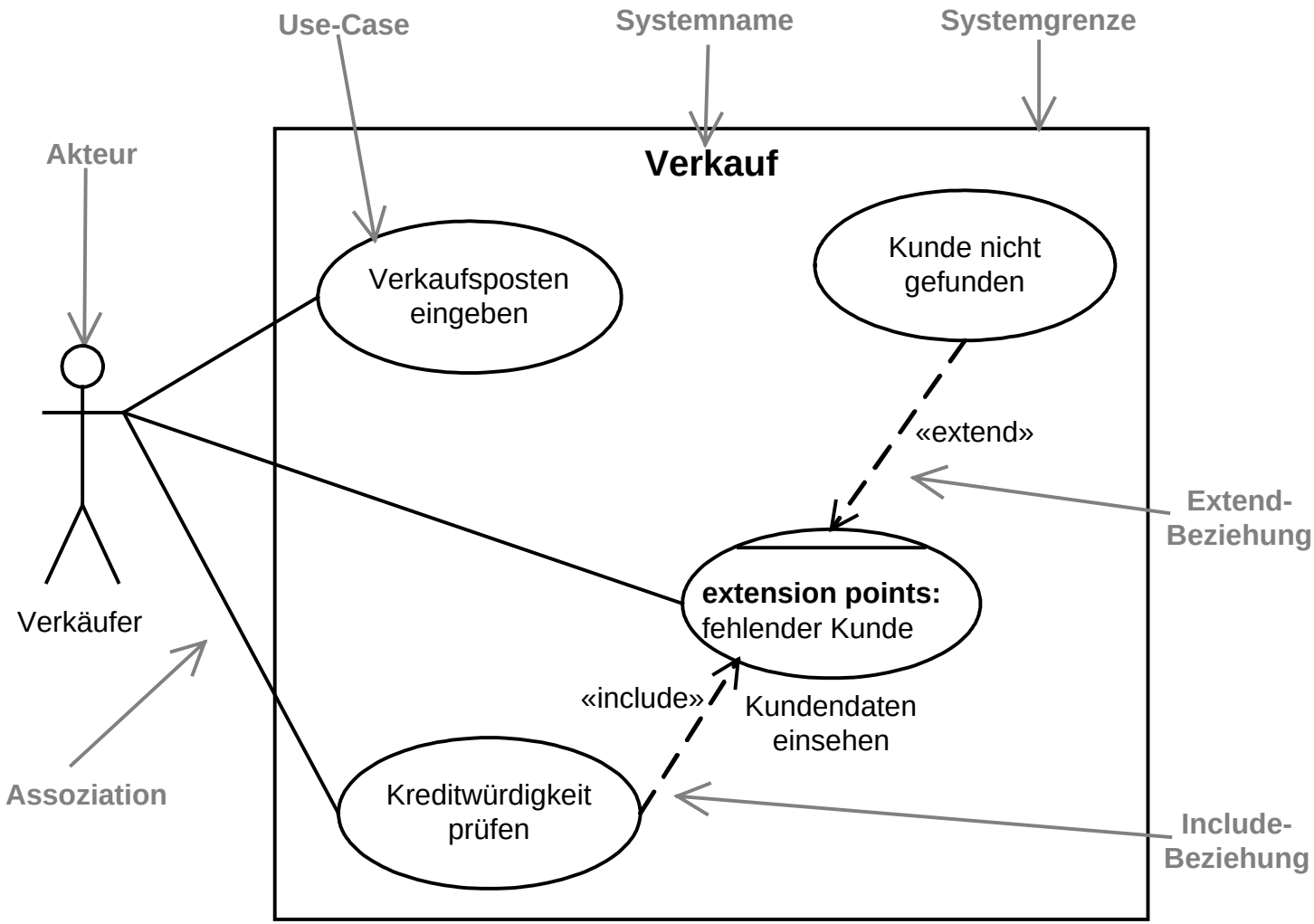
- **Use-Case-Diagramm**
  Darstellung der funktionalen Dienstleistungen eines Systems
- **Aktivitätsdiagramm**
  Visualisiert mögliche Abläufe mit veränderbarem Detaillierungsgrad
- **Zustandsautomat**
  Abbilden des Verhaltens auf Zustände und Zustandsübergänge
- **Interaktionsmodellierung**
  Stellt den Nachrichten- und Datenaustausch zwischen zwei Kommunikationspartnern dar
  - Sequenzdiagramm
  - Kommunikationsdiagramm
  - Timing-Diagramm
  - Interaktionsübersichtsdiagramm

Use-Case-Diagramm

- **Funktion**:
  Darstellung der funktionalen Dienstleistungen eines Systems

- **Aufgabe im Projekt**:
  - Eine erste konzeptuelle Darstellung zur Abgrenzung des Systems gegenüber seiner Umwelt
  - Schaffen einer Kommunikationsgrundlage zwischen Stakeholdern
  - Erstellung planbarer Einheiten

Use-Case-Diagramm



Use-Case-Diagramm

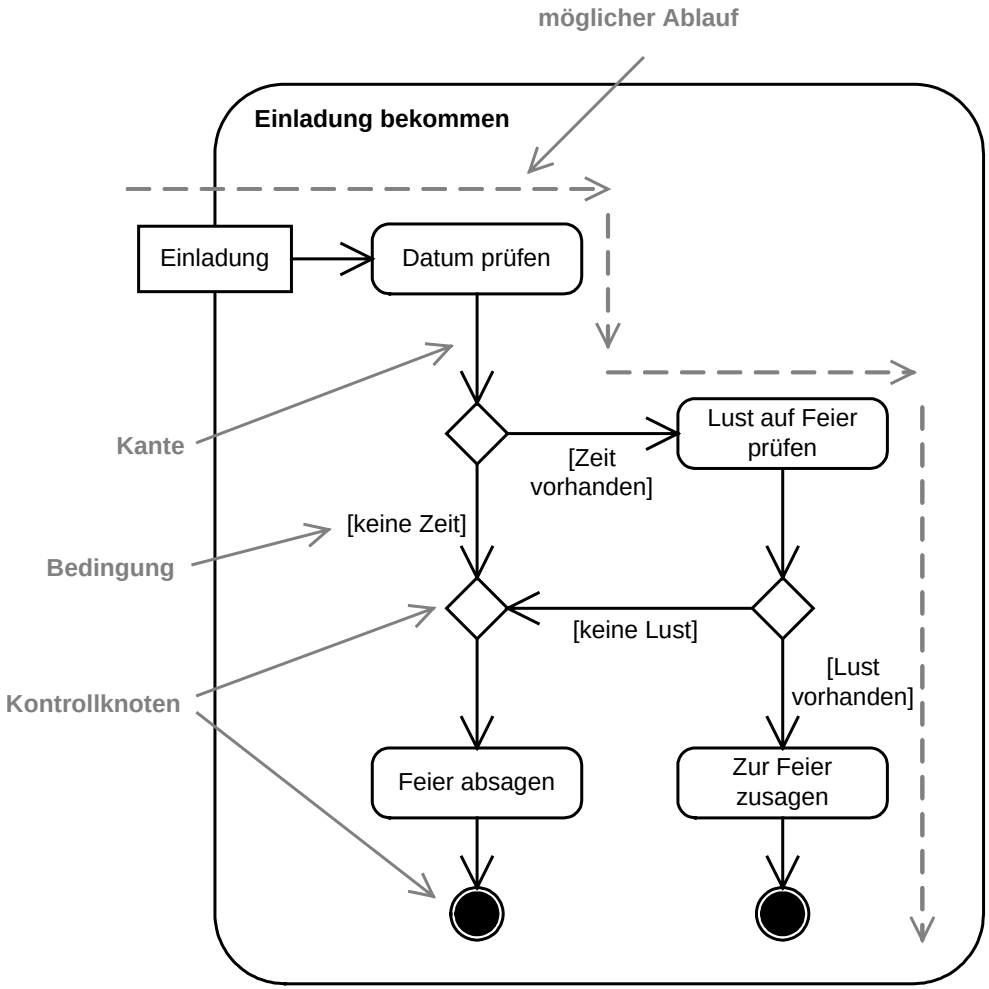
- Geändert in UML 2:
  - Name eines Akteurs nicht mehr optional, sondern verpflichtend
  - Classifier können Use-Cases besitzen, nicht nur Pakete
  - Vorbedingung und extension point werden als Notiz an die Extend-Beziehung angehängt

Aktivitätsdiagramm

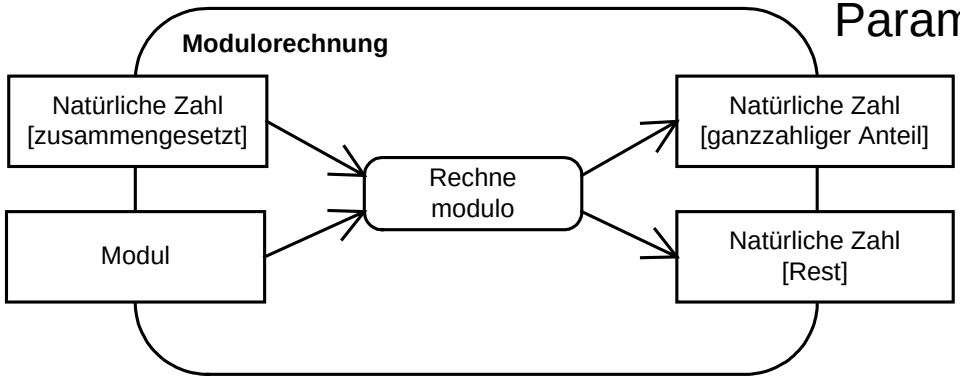
- **Funktion**:
  Visualisiert mögliche Abläufe mit veränderbarem Detaillierungsgrad

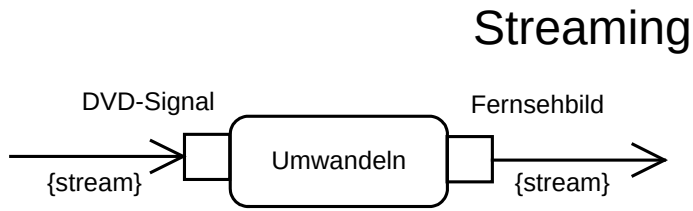
- **Aufgabe im Projekt**:
  - Geschäftsprozessmodellierung
  - Beschreibung von Use-Cases
  - Implementieren von Operationen

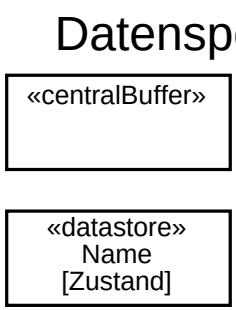
Aktivitätsdiagramm

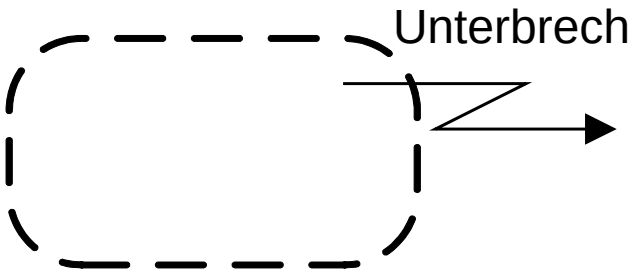


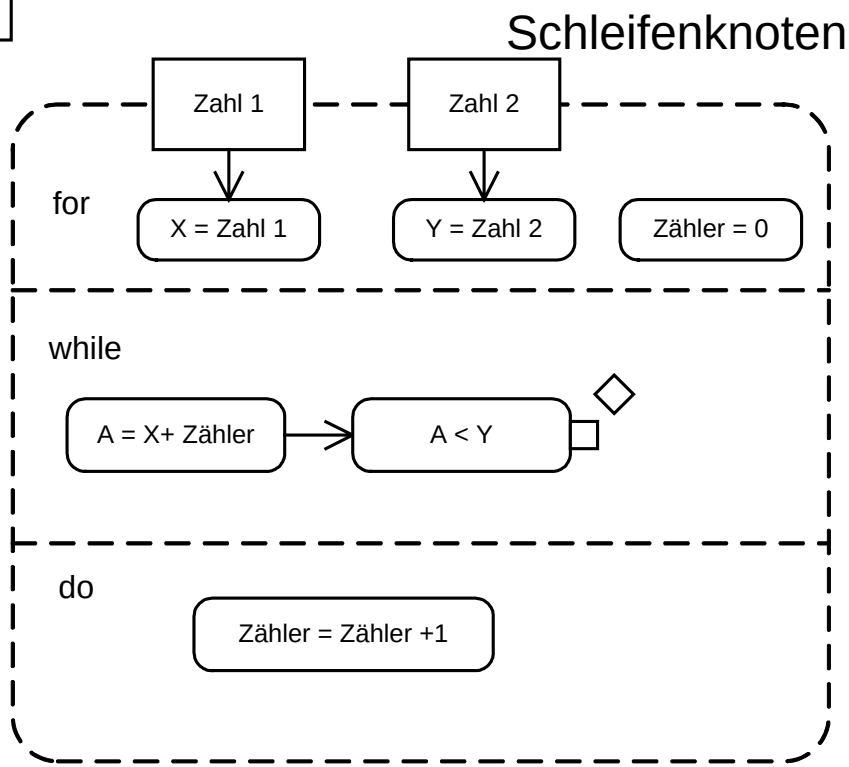
Aktivitätsdiagramm



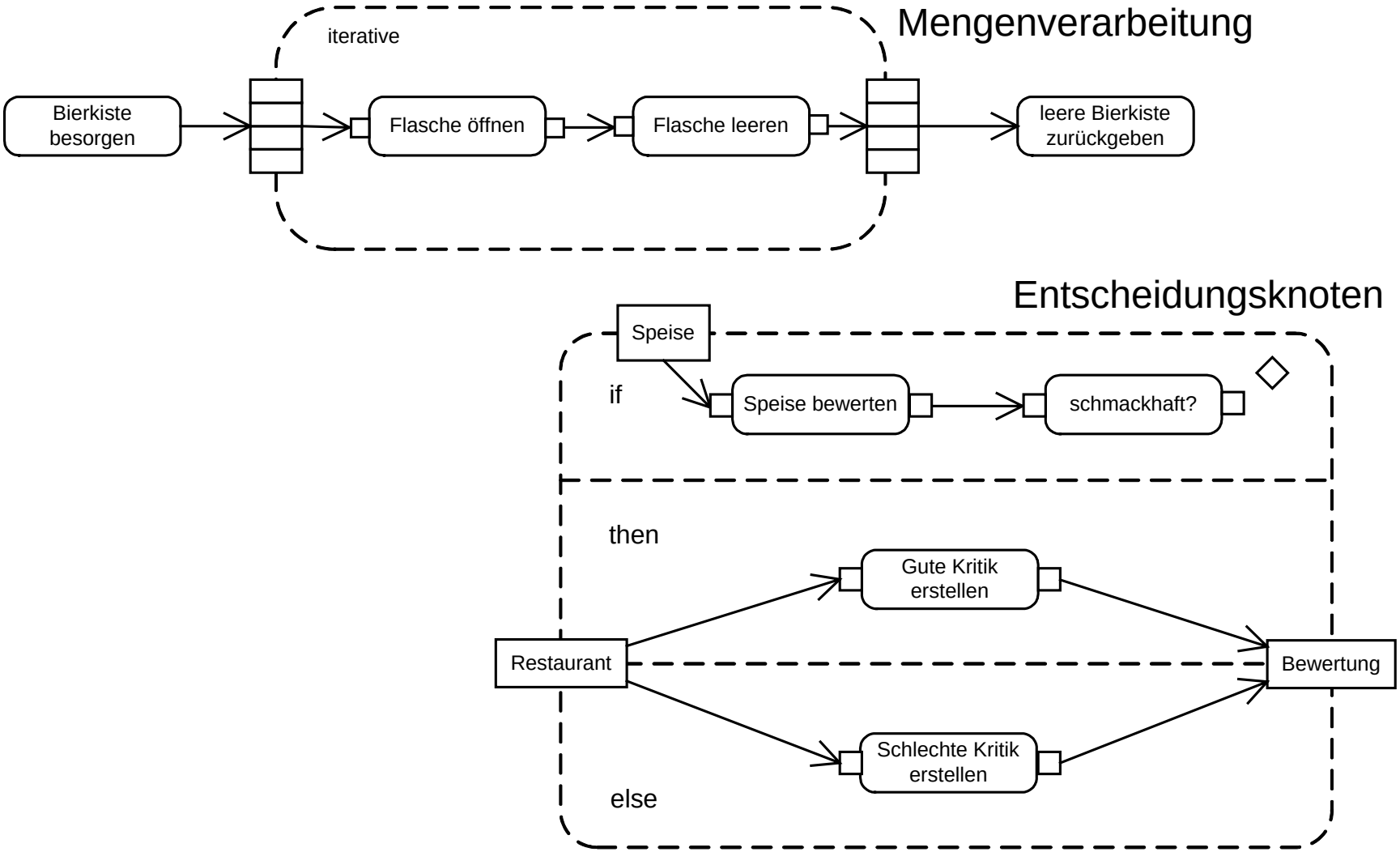








Aktivitätsdiagramm



Aktivitätsdiagramm

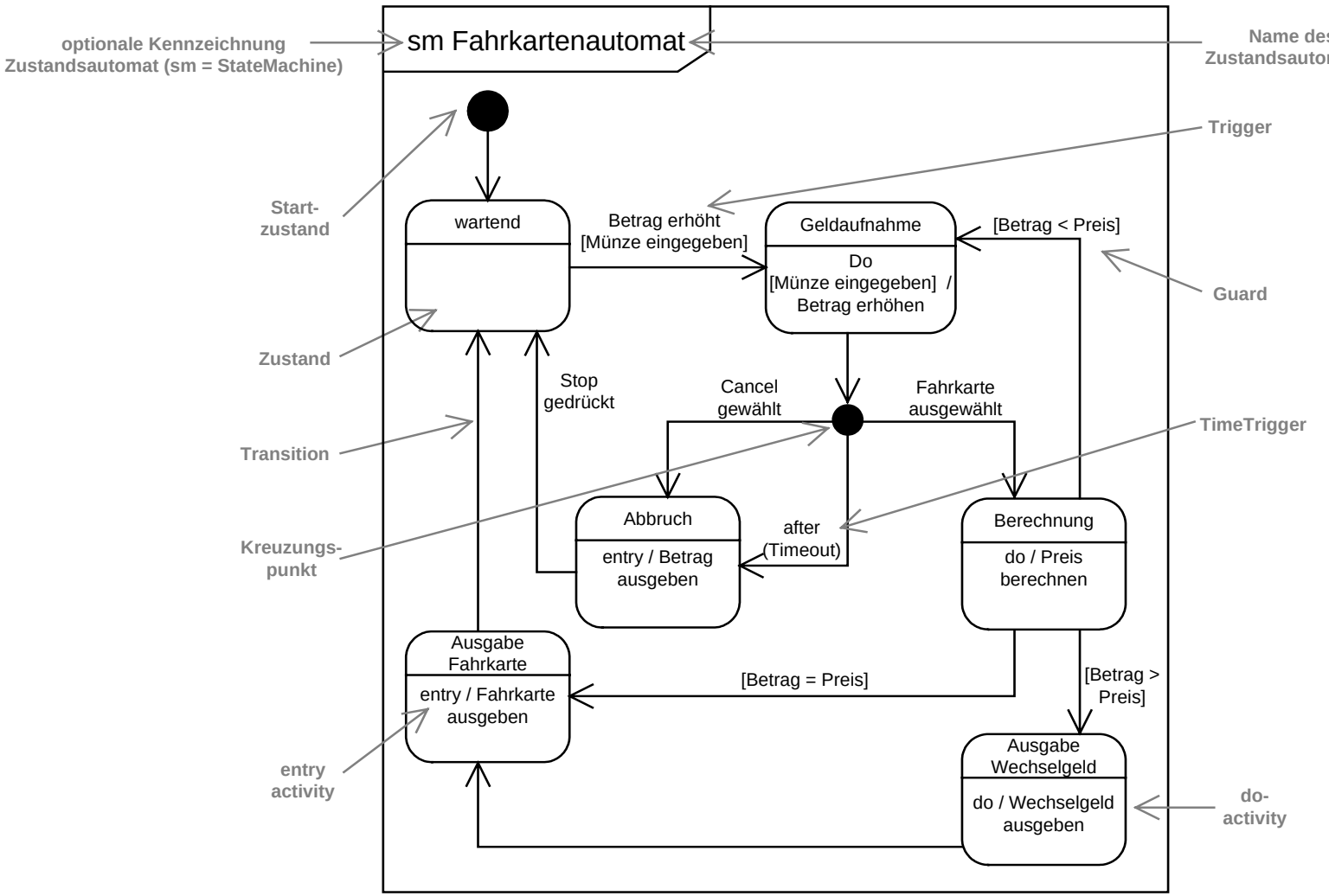
- Neu in UML 2:
  - **Neue Notationselemente:** Strukturierte Knoten, Entscheidungsknoten, Schleifenknoten, Mengenverarbeitungsknoten, Unterbrechungsbereich, Datenspeicher und Bufferknoten, Parametersatz
  - Aktivitäten verwenden eine den Petri-Netzen ähnlich Semantik (Token-Konzept)
  - Aktivitäten können Ein- und Ausgabeparameter enthalten
  - Aktionen können mit Vor- und Nachbedingungen verknüpft werden
- Geändert in UML 2:
  - Aktivitäten sind nun unabhängig von Zustandsautomaten
  - Es sind nun mehrere Startknoten erlaubt
  - Parallele Abläufe müssen nicht wieder zusammengeführt werden
  - Aktivitätsbereiche können hierarchisch oder multidimensional sein
  - Die Notation von Aktionen entspricht der Notation von Zuständen der UML 1.x

Zustandsautomat

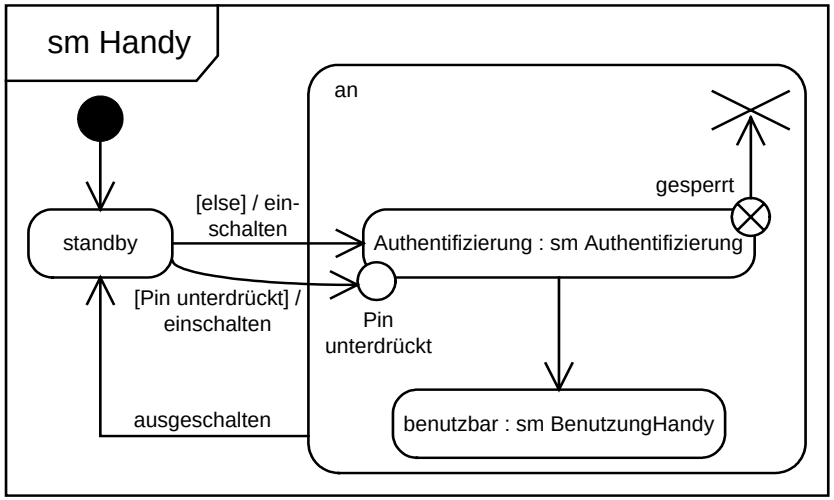
- **Funktion**:
  Abbilden des Verhaltens auf Zustände und Zustandsübergänge

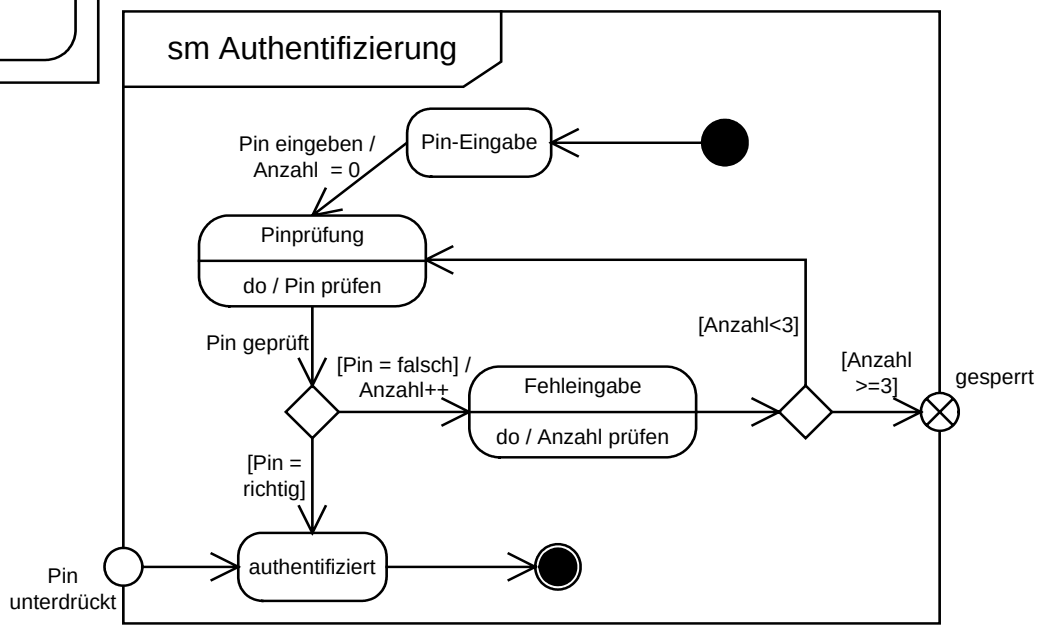
- **Aufgabe im Projekt**:
  - Zustandsbeschreibung von Classifiern, ...
  - Detaillierung von Use-Cases
  - Verhaltensbeschreibung von Interfaces und Ports
  - Detailbeschreibung von Signal- und Ereignisempfang

Zustandsautomat



Zustandsautomat





Zustandsautomat

- Neu in UML 2:
  - Ports können nun Protokollzustandsautomaten besitzen
  - Ein-, Austrittspunkte und Terminatoren wurden eingeführt
  - Regeln zur Ergänzung und Ersetzung von Transitionen bei vererbten Zustandsautomaten wurde hinzugefügt
- Geändert in UML 2:
  - Tiefe Historien können auch Ziel einer Transition innerhalb des enthaltenen Zustands sein (also nicht nur von außen)
  - Das vererbte Verhalten wird mit einem Zustandsautomaten dargestellt und nicht nur durch eine Notiz

Interaktionsmodellierung

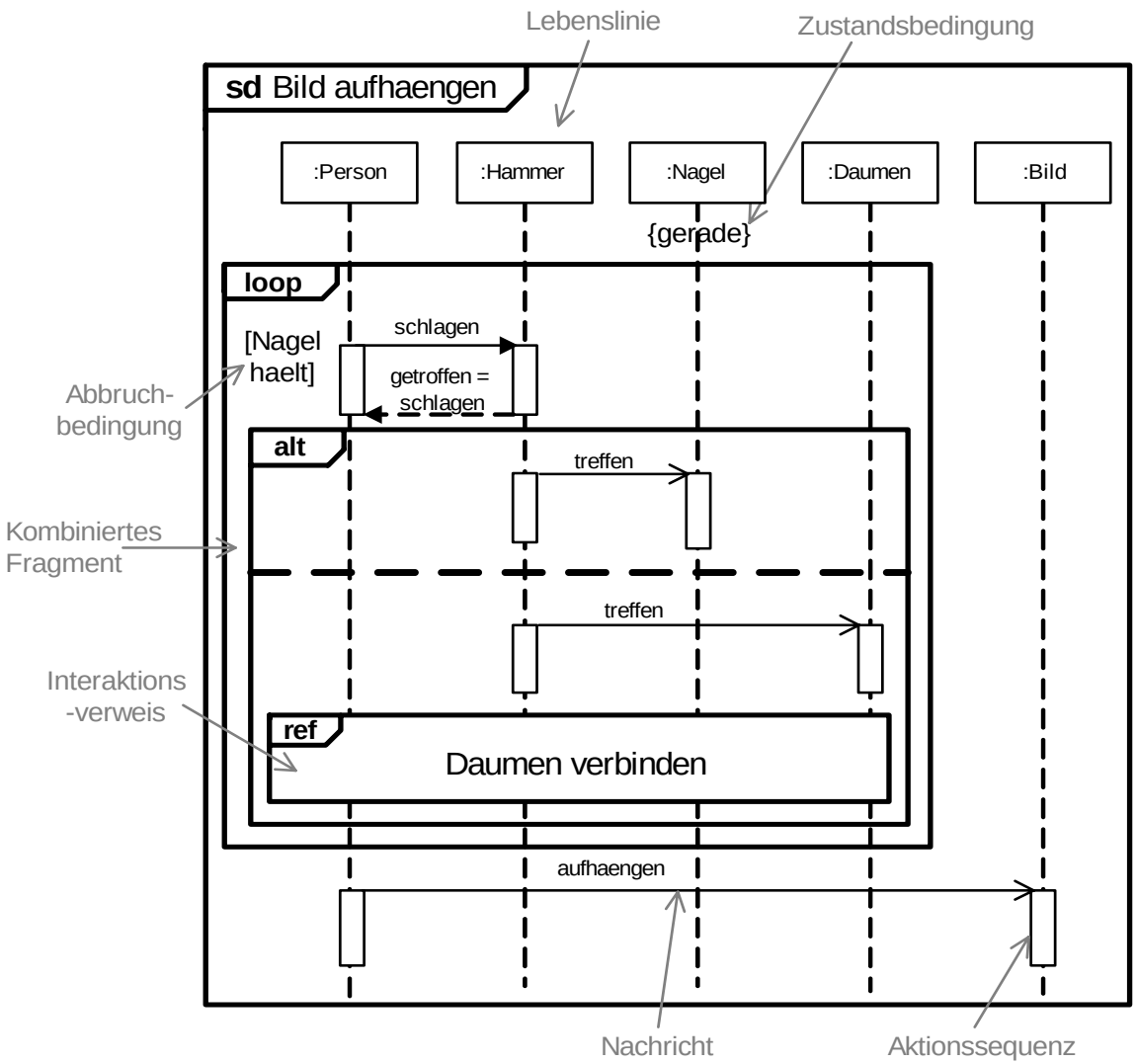
- **Funktion**:
  - Interaktion ist der Nachrichten- und Datenaustausch zwischen zwei Kommunikationspartnern
  - Nachrichtenaustausch verbindet Sende- und Empfangsereignisse

- **Aufgabe im Projekt**:
  - Interaktionsdiagramme zeigen nur Sichten auf Interaktionen
  - Diagrammwahl nach hervorzuhebenden Modellierungsaspekt

Interaktionsmodellierung

- Diagrammwahl / Modellierungsaspekt :
  - Sequenzdiagramm:
    - + Die Reihenfolge, in der der Nachrichtenaustausch stattfindet
  - Kommunikationsdiagramm:
    - + Das Zusammenspiel von **strukturierten** Kommunikationspartnern
    - + Modellierung von Prinzipien und Konzepten
  - Timing-Diagramm:
    - + Darstellung der zeitlichen Veränderung eines Classifiers
    - + Modellierung zeitkritischer Zustands- und Wertänderungen sinnvoll
  - Interaktionsübersichtsdiagramm:
    - + Darstellung der Ablaufreihenfolge mehrerer Interaktionen
    - + Logischer Zusammenhang zwischen Interaktionsdiagrammen
    - + Brückenschlag zwischen Aktivitäts- und Interaktionsdiagrammen

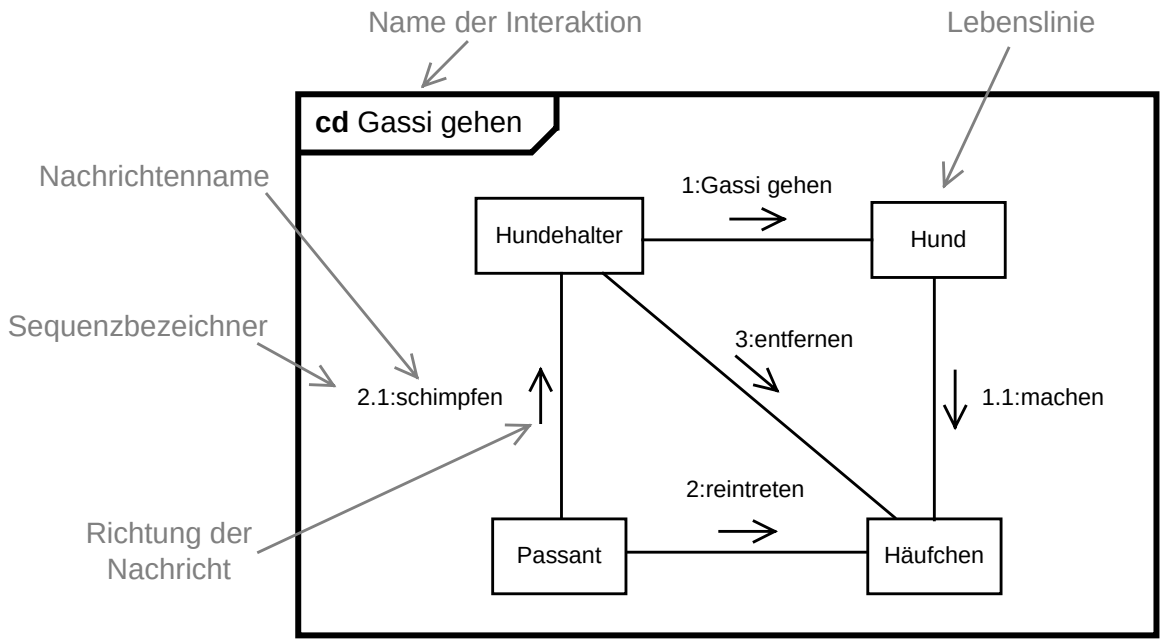
Sequenzdiagramme



Sequenzdiagramme

- Neu in UML 2:
  - Zustandsinvarianten können an Lebenslinien angetragen werden
  - „lost“ und „found“ Nachrichten eingeführt
- Geändert in UML 2:
  - Innerhalb von Sequenzdiagrammen kann auf andere Interaktionen verwiesen werden (Zerlegung von Abläufen oder Lebenslinien)
  - Alle Kontrollflussmöglichkeiten (if, switch, ...) höherer Programmiersprachen durch kombinierte Fragmente, sehr gute Unterstützung von nebenläufiger Modellierung
- Entfällt in UML 2:
  - Nachrichtenart „unspezifiziert“ entfällt

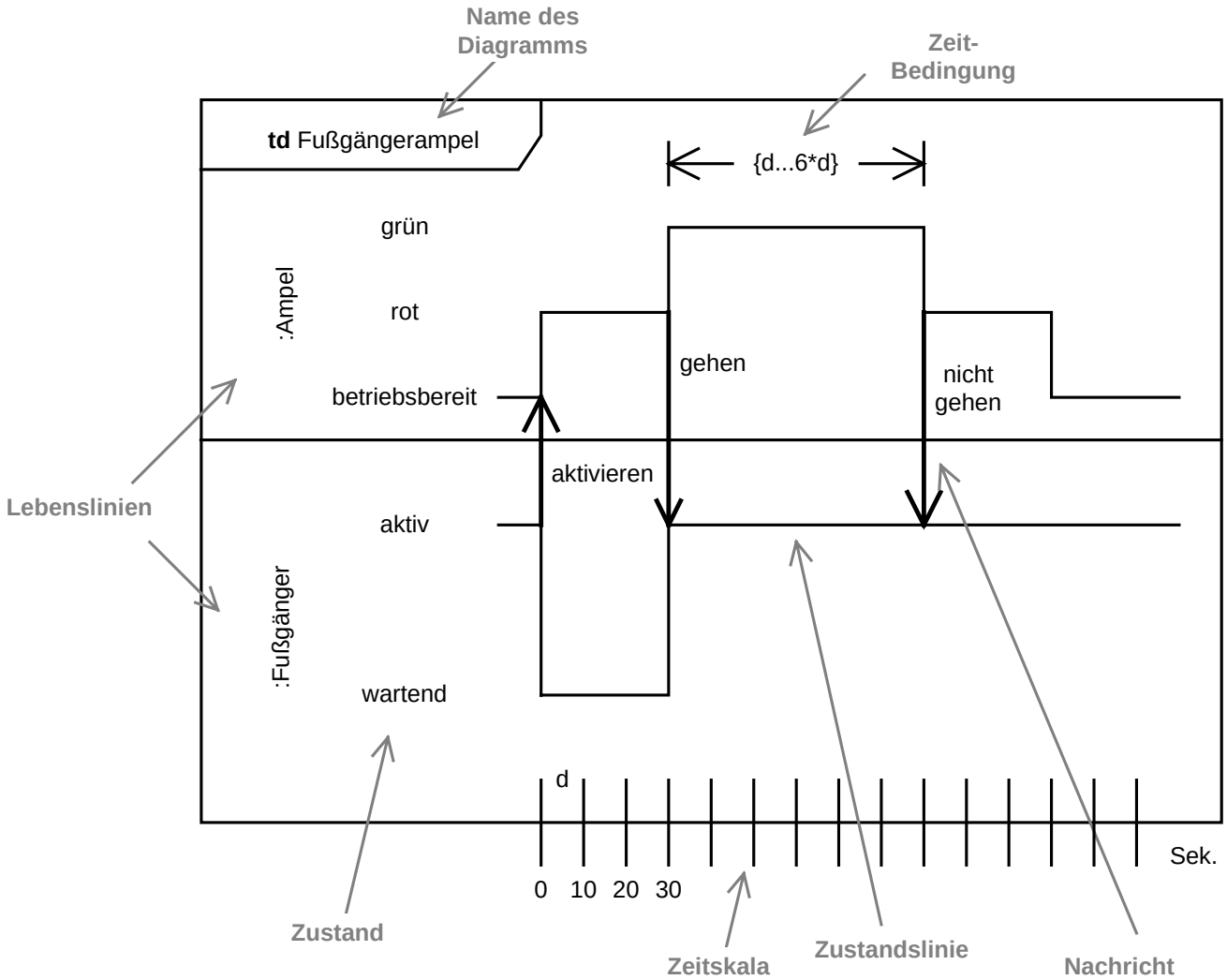
Kommunikationsdiagramme



Kommunikationsdiagramme

- Geändert in UML 2:
  - Kommunikationsdiagramm statt Kollaborationsdiagramm
  - Subset des Sequenzdiagramms, z. B.
    - keine Interaktionsverweise („ref“)
    - keine kombinierten Fragmente
    - Ereignisreihenfolge wird ignoriert

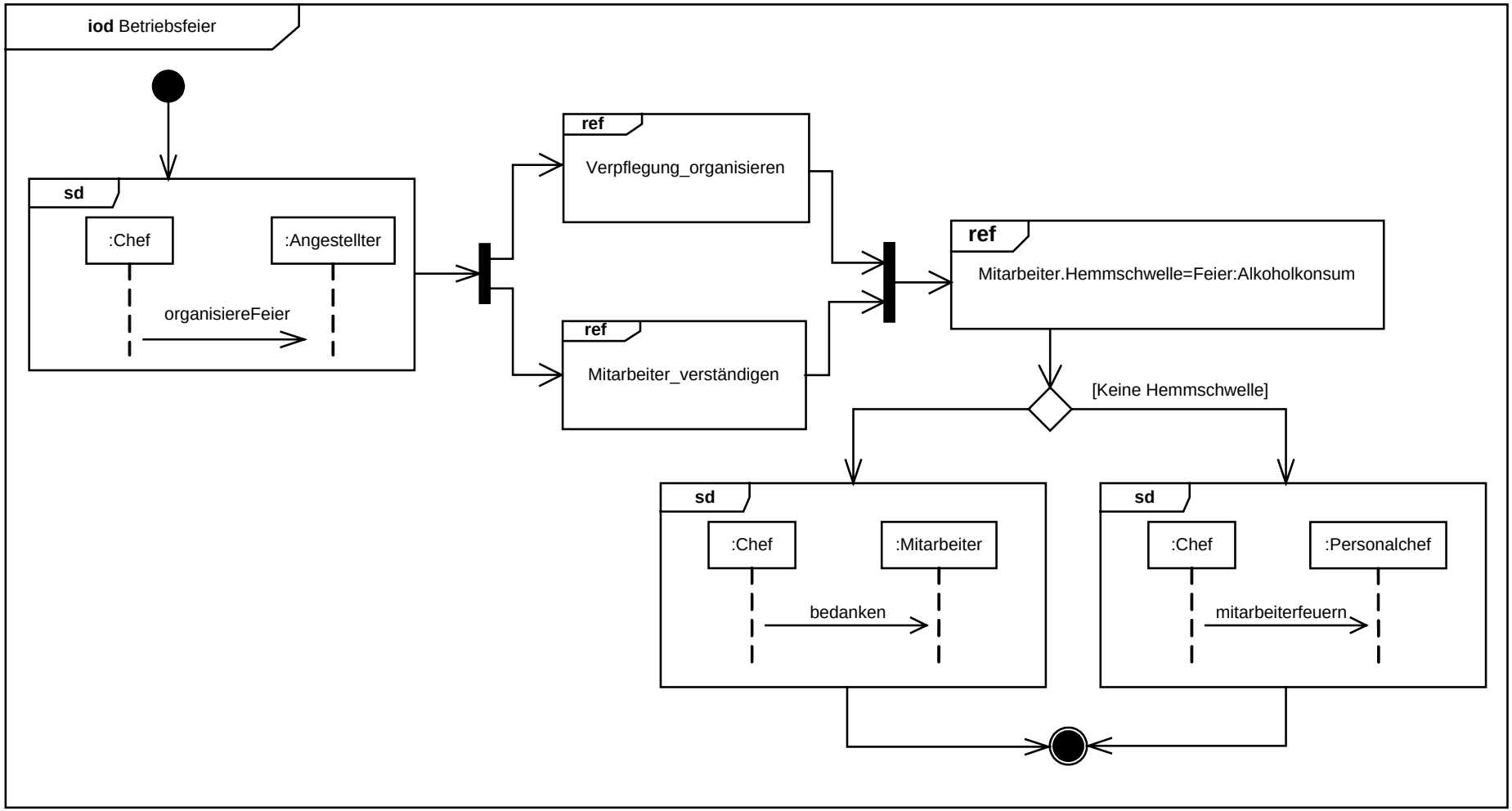
Timing-Diagramm



Timing-Diagramm

- Neu in UML 2:
  - Komplett neu in die UML 2 eingeführt
  - Bereits seit Jahren in Elektrotechnik genutzt (z. B. für elektronische Schaltvorgänge)

Interaktionsübersichtsdiagramme



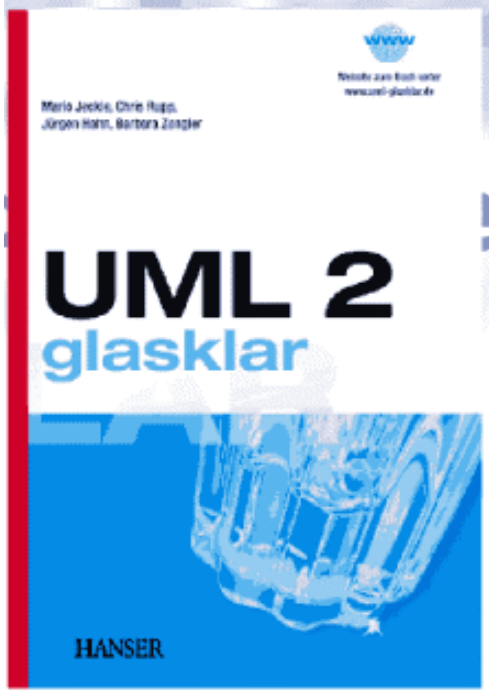
Interaktionsübersichtsdiagramme

- Neu in UML 2:
  - Komplett neu in die UML 2 eingeführt

Fazit --- dynamische Diagramme in UML 2

- Viel neu - viel geklaut
- Aktivitätsdiagramme und Sequenzdiagramme faktisch neu
- Verbesserte Unterstützung der Codeabbildung
- Bessere Modellierung von Nebenläufigkeiten und Zeitverhalten möglich ➔ dennoch keine „Echtzeitsprache“
- Deutlich bessere Schnittstelle zu Strukturdiagrammen
- Notationsvielfalt erfordert Tailoring
- Interaktionsübersichtsdiagramm, Timingdiagramm erfordern Nachbesserungen

Damit Sie klar sehen!



damit Sie klar sehen!

Trainings: **UML 2 Update:** **27.11.2003**
OOA: **09.12.-10.12.2003**
OOD: **11.12.-12.12.2003**

Beratung: wann immer Sie wollen ;-))

Infos:

- www.uml-glasklar.de
- www.jeckle.de
- www.sophist.de