



Harald Störrle

UML 2 für Studenten

Mit UML-Syntax-Poster

UML 2 für Studenten

UML 2 für Studenten

Inhaltsverzeichnis

UML 2 für Studenten

Inhaltsverzeichnis

Vorwort

Teil I Einführung

Kapitel 1 UML (nicht nur) für Studenten

Kapitel 2 UML im Überblick

Kapitel 3 UML im Kontext

Kapitel 4 UML in der Praxis

Teil II Struktur

Kapitel 5 Klassen und Beziehungen

Kapitel 6 Architektur und Komponenten

Kapitel 7 Nichtfunktionale Anforderungen

Kapitel 8 Object Constraint Language

Teil III Verhalten

Kapitel 9 Nutzfälle

Kapitel 10 Zustandsautomaten

Kapitel 11 Aktivitäten

Kapitel 12 Interaktionen

Teil IV Anhänge

Anhang A UML-Syntax

Anhang B UML Diagramme und Modelle

Anhang C UML Erweiterbarkeit

Anhang D UML Metamodell

Anhang E Glossar

Literaturverzeichnis

Inhaltsverzeichnis

Sachregister

Vorwort

Teil I Einführung

UML (nicht nur) für Studenten

UML im Überblick

UML im Kontext

UML in der Praxis

Teil II Struktur

Klassen und Beziehungen

Architektur und Komponenten

Nichtfunktionale Anforderungen

Object Constraint Language

Teil III Verhalten

Nutzfälle

Zustandsautomaten

Aktivitäten

Interaktionen

Teil IV Anhänge

Anhang A UML-Syntax

A.1 Grafische Notationselemente

A.1.1 Allgemein

A.1.2 Klassendiagramme 1 Klassen und Objekte

A.1.3 Klassendiagramme 2 ungerichtete Beziehungen

A.1.4 Klassendiagramme 3 gerichtete Beziehungen

A.1.5 Montagediagramme

A.1.6 Paketdiagramme

A.1.7 Verteilungsdiagramme

A.1.8 Nutzfalldiagramme

A.1.9 Zustandsdiagramme

Inhaltsverzeichnis

A.1.10 Aktivitätsdiagramme

A.1.11 Interaktionsdiagramme

A.2 Anschriften

Klassen und Assoziationen

Pakete

OCL

Zustandsautomaten

Interaktionen

A.3 Eindeutige Bezeichner

A.4 Namen

Systeme

Domänen

Prozesse, Ereignisse

Funktionen, Dienste, Aktivitäten

Operationen

Zustände

Zustandsautomaten

Klassen, Attribute, Pakete, Komponenten

Zulässige Zeichen

A.5 Layout

Form follows function

Beschränkung auf das Wesentliche

Leserichtung beachten

Gleiches gleich darstellen

Anhang B UML Diagramme und Modelle

Klassendiagramme

Architekturdiagramme

Kollaborationen

Installationsdiagramme

Nutzfalldiagramme

Zustandsautomaten

Aktivitätsdiagramme

Interaktionsdiagramme

Inhaltsverzeichnis

Anhang C UML Erweiterbarkeit

C.1 Stereotype

C.2 Tagged Values

C.3 Profile

Anhang D UML Metamodell

Anhang E Glossar

Literaturverzeichnis

Sachregister

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

Inhaltsverzeichnis

U

V

W

X

Z

Ins Internet: Weitere Infos zum Buch, Downloads, etc.

© Copyright

UML 2.0 Syntax Poster

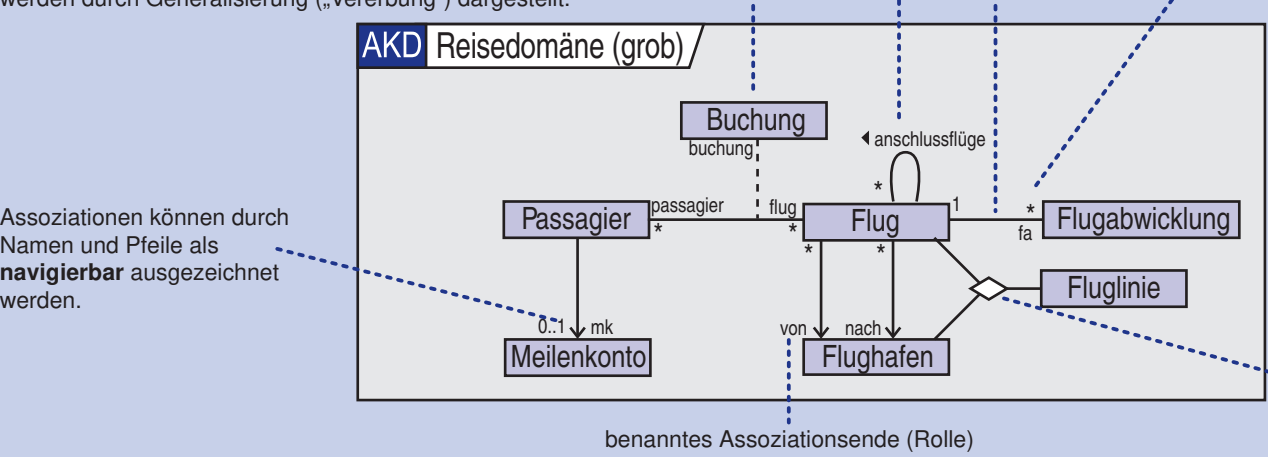
UML 2.0 Syntax

© Harald Större, www.pst.ifi.lmu.de/~stoerre

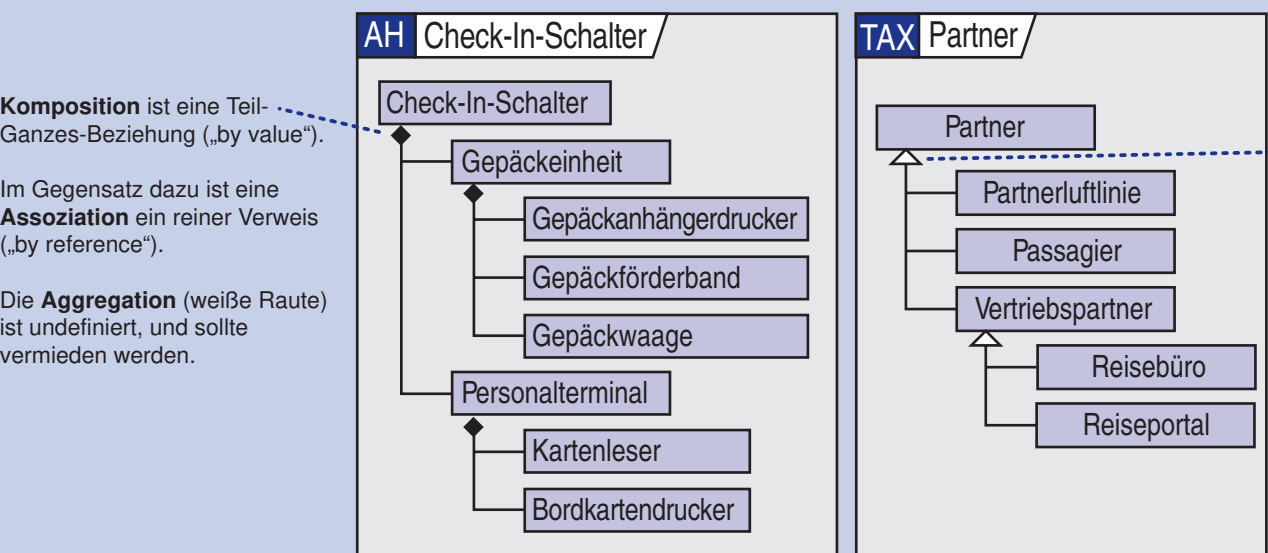


Klassendiagramme

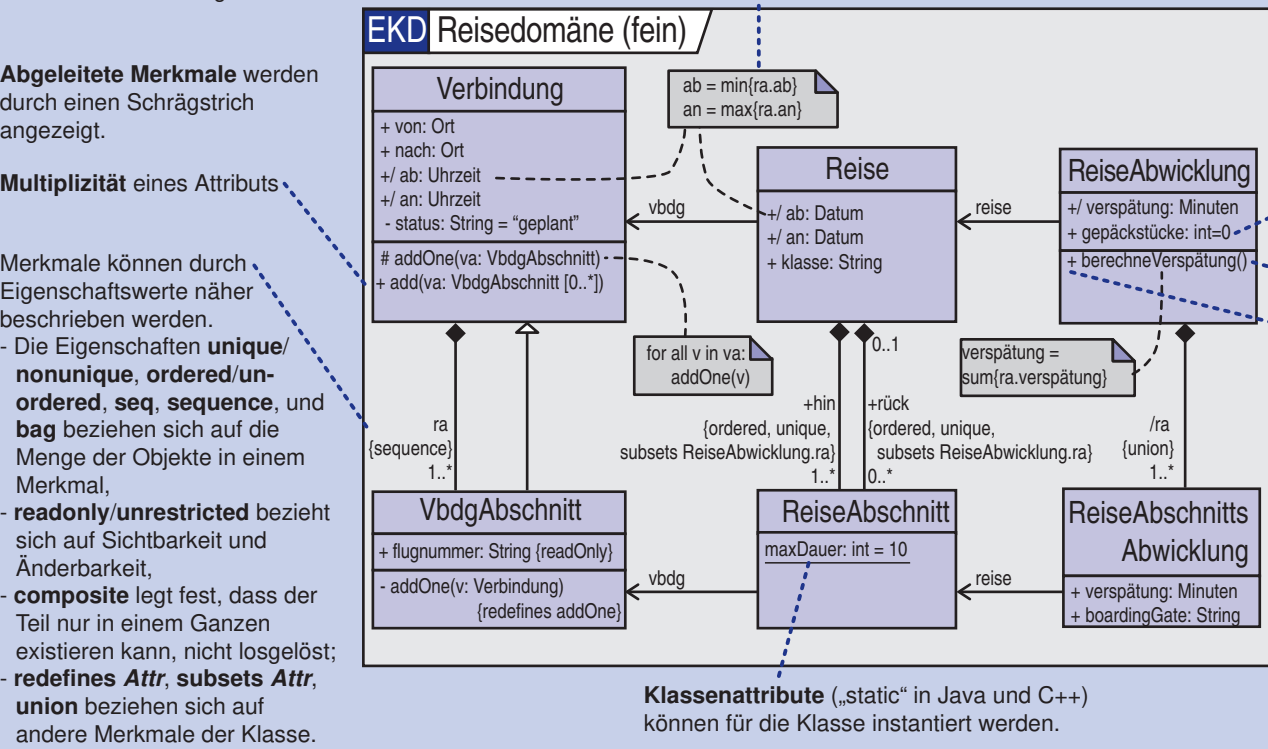
Analyse-Klassendiagramme stellen Geflechte von Konzepten dar, wobei Konzepte durch Klassen und ihre Beziehungen durch Assoziationen dargestellt werden. Verallgemeinerung und Spezialisierung von Konzepten werden durch Generalisierung („Vererbung“) dargestellt.



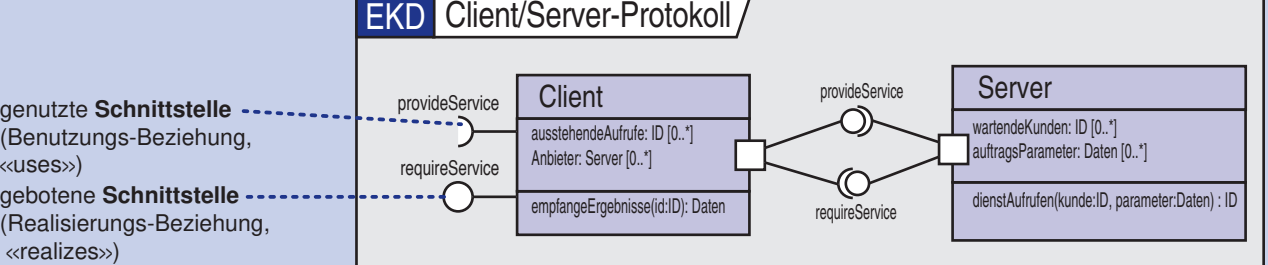
Mit einer **Aggregationshierarchie** kann der statische Aufbau verdeutlicht werden.



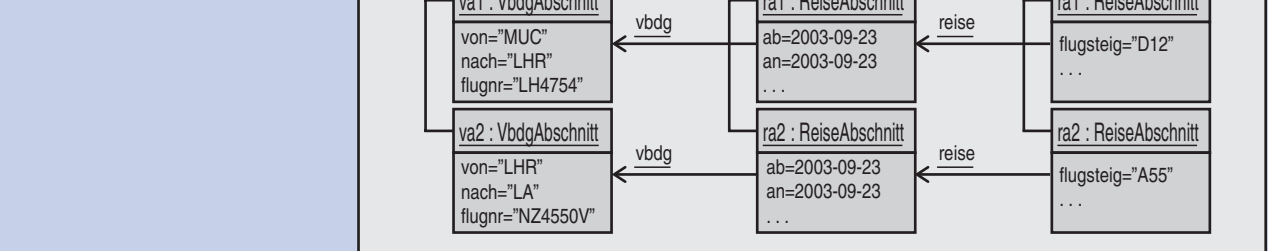
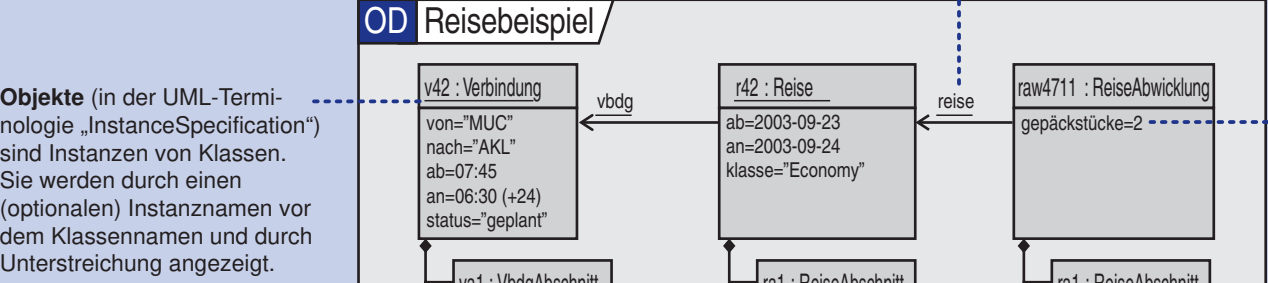
Entwurf-Klassendiagramme stellen den Übergang von einem Analysemodell zu einem Implementationsmodell dar und dokumentieren Technologie-unabhängige Entwurfsentscheidungen.



Klassen und Anschlüsse können durch **Schnittstellen** näher beschrieben werden. Dabei kann zwischen gebotenen und genutzten Schnittstellen unterschieden werden.

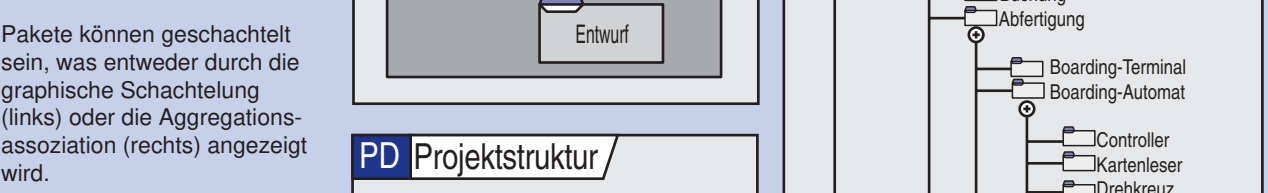
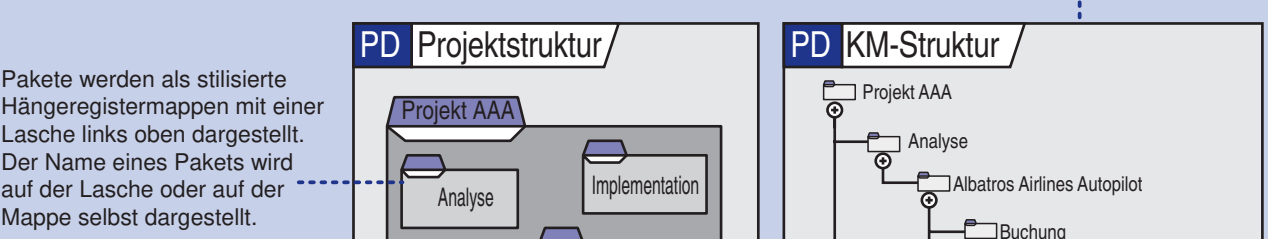


Mit **Objektdiagrammen** lassen sich mögliche Instanzierungen von Klassendiagrammen darstellen, also quasi „Momentaufnahmen“, und damit Systemzustände.

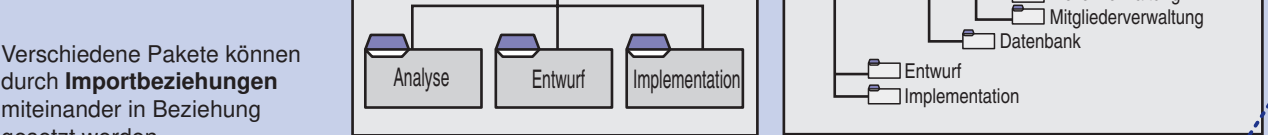


Paketdiagramme

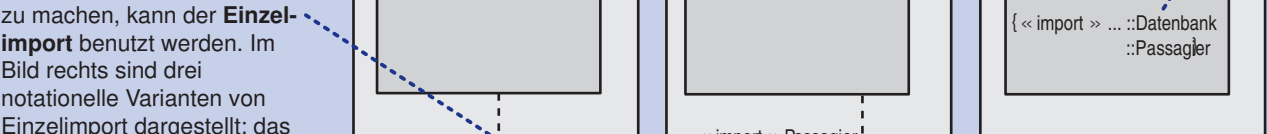
Paketdiagramme stellen die Gruppierung von Modellelementen zu Namensräumen dar. Paketdiagramme werden gleitend zur Beschreibung von Projektstrukturen und Software verwendet.



Die **Aggregationsbeziehung** für Pakete ähnelt der Komposition von Klassen.



Verschiedene Pakete können durch **Importbeziehungen** miteinander in Beziehung gesetzt werden.

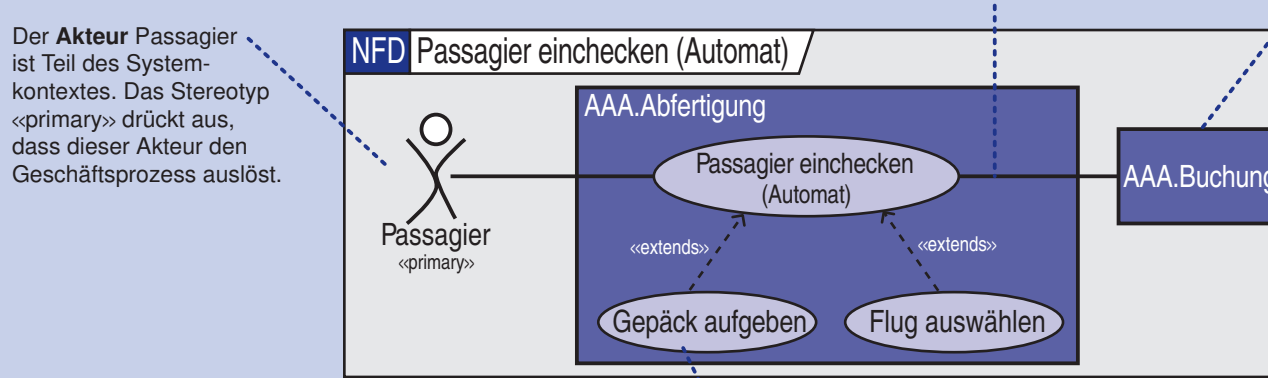


Um gezielt einzelne Elemente (z.B. Klassen oder Pakete) sichtbar (und damit zugreifbar) zu machen, kann der **Einzel-Import** benutzt werden. Im Bild rechts sind drei notatorische Varianten von Einzelimport dargestellt: das Paket, den Namen des Pakets, das Element, das auf das Element zugegriffen werden soll.

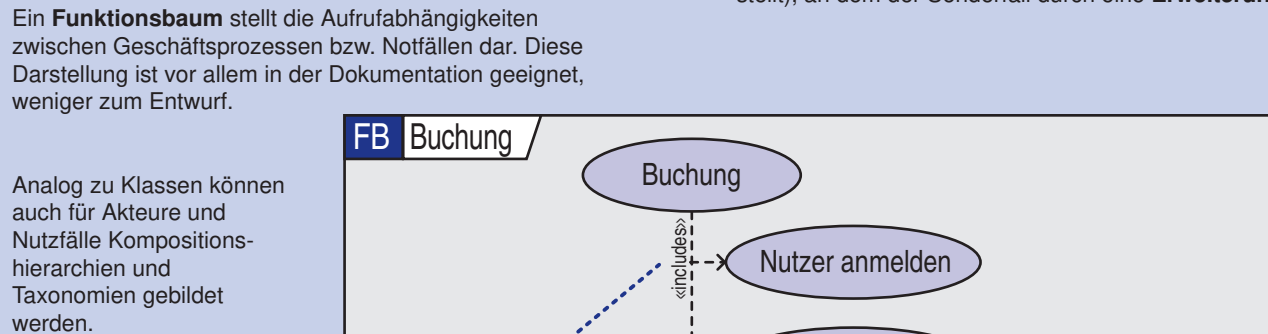
Soll ein Element nicht weiter exportiert werden dürfen, wird es mit «**access**» importiert.

Nutzfalldiagramme

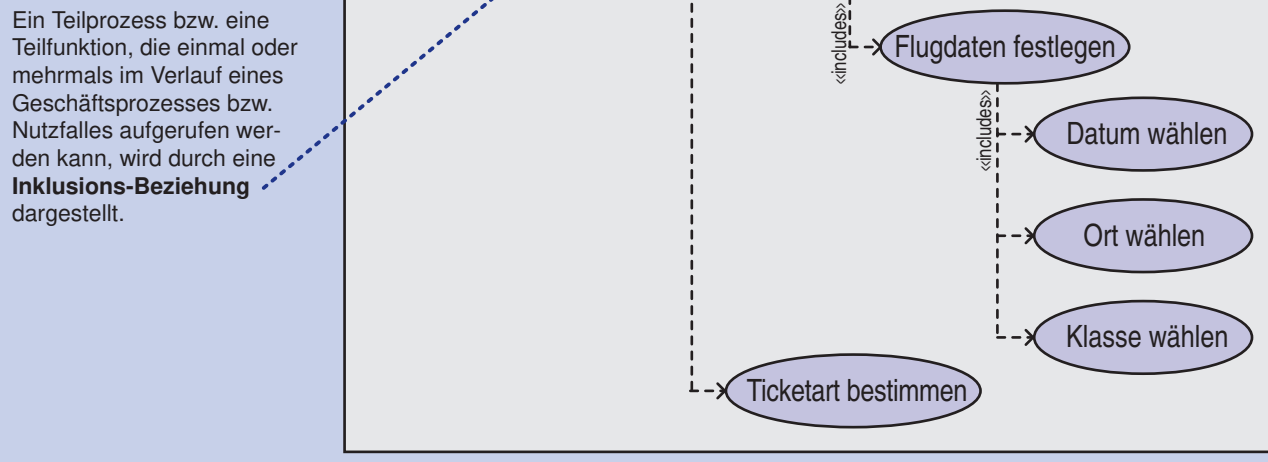
Nutzfalldiagramme sind vor allem als Übersichtsdarstellungen nützlich. Sie stellen einseitig Elemente eines Systemkontextes dar (also Akteure und Nachbarsysteme), andererseits Geschäftsprozesse bzw. Nutzfälle des Systems. Daneben können auch Abhängigkeiten zwischen Geschäftsprozessen bzw. Nutzfällen mit Inklusions- und Erweiterungsbeziehungen angezeigt werden.



Ein **Funktionsbaum** stellt die Aufrufabhängigkeiten zwischen Geschäftsprozessen bzw. Nutzfällen dar. Diese Darstellung ist vor allem in der Dokumentation geeignet, weniger zum Entwurf.

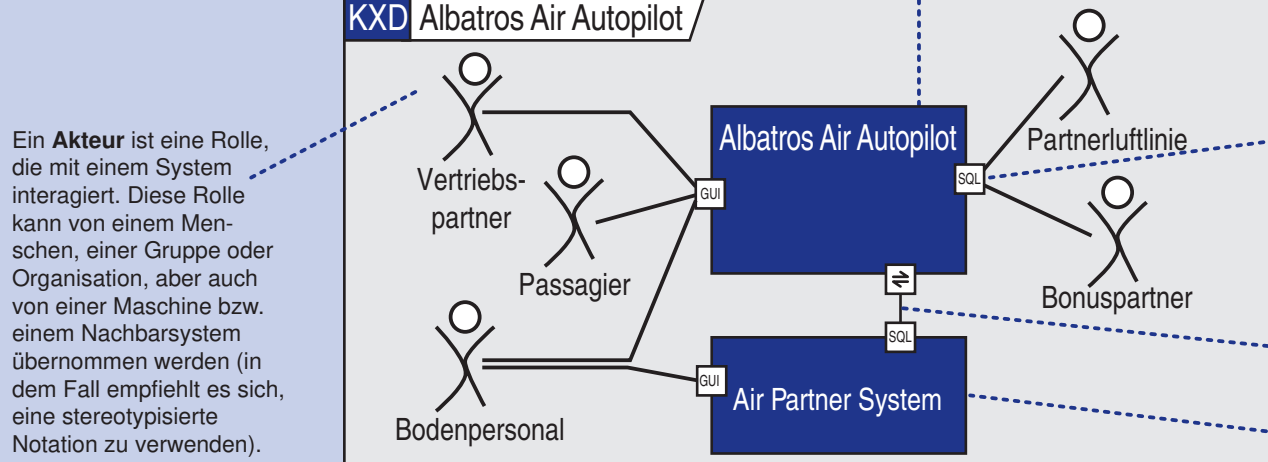


Analog zu Klassen können auch für Akteure und Nutzfälle Kompositionshierarchien und Taxonomie gebildet werden.



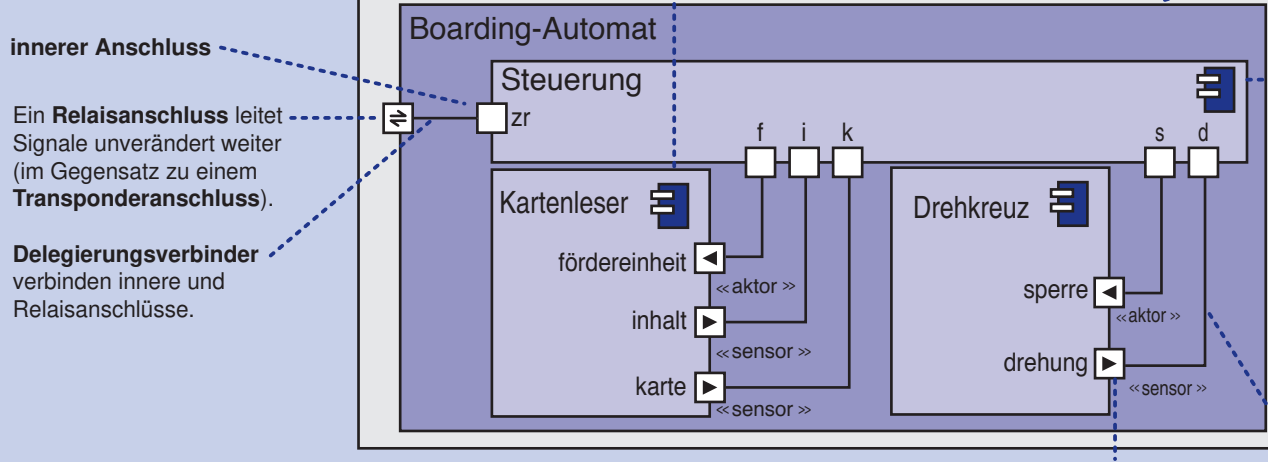
Montagediagramme

Ein **Kontextdiagramm** dient dazu, die Umgebung eines Systems zu definieren, und so das System abzugrenzen.



Ein **Akteur** ist eine Rolle, die mit einem System interagiert. Diese Rolle kann von einem Menschen, einer Gruppe oder Organisation, aber auch von einer Maschine bzw. einem Nachbarsystem übernommen werden (in dem Fall empfiehlt es sich, eine stereotypisierte Notation zu verwenden).

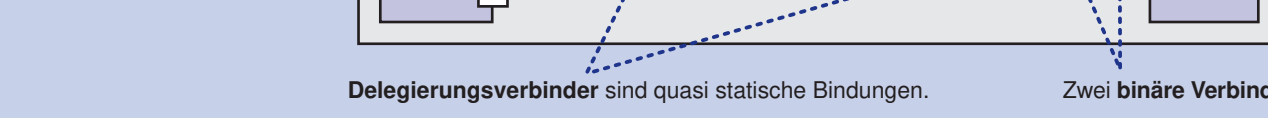
Mit einem **Montagediagramm** kann die Grobstruktur eines Systems beschrieben werden. In dieser Verwendung lassen sich auch Hardware-nahe Strukturen gut darstellen (so begründet sich z.B. auch der Name „Anschluss“).



Auf einem noch feineren Detailgrad können **Montagediagramme** dazu benutzt werden, die Struktur zusammengesetzter Klassen zu beschreiben.

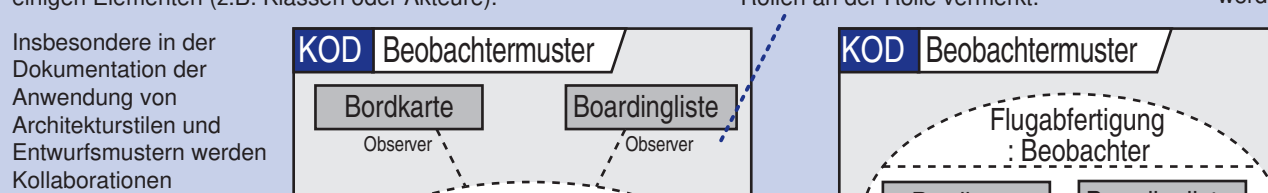


Ein **ternärer Verbinder** verbindet innere und Relaisanschlüsse.

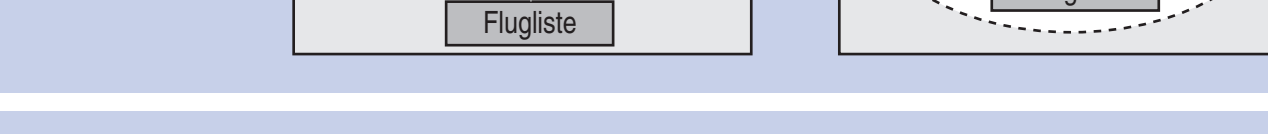


Kollaborationen

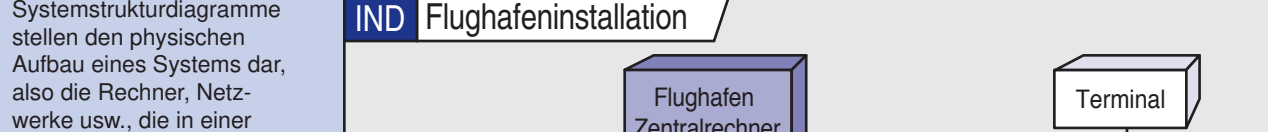
Kollaborationen definieren einen strukturellen Zusammenhang und eine Verteilung von Rollen zwischen einzelnen Elementen (z.B. Klassen oder Akteure).



Alternativ können die Rollen innerhalb der Kollaboration angezeigt werden, verbunden, ggf. werden die Namen der Rollen an der Rolle vermerkt.



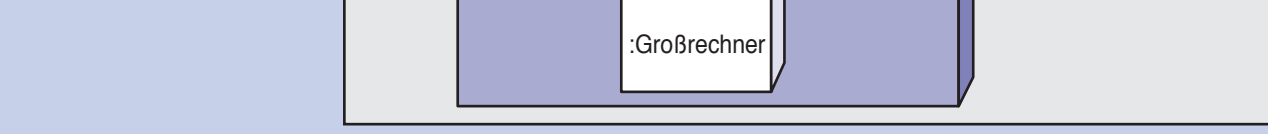
Ein **Knoten** repräsentiert einen Rechner oder eine sonstige aktive Komponente in einer Installation. Knoten können ineinander geschachtelt werden.



Ein **Knoten** repräsentiert einen Rechner oder eine sonstige aktive Komponente in einer Installation. Knoten können ineinander geschachtelt werden.

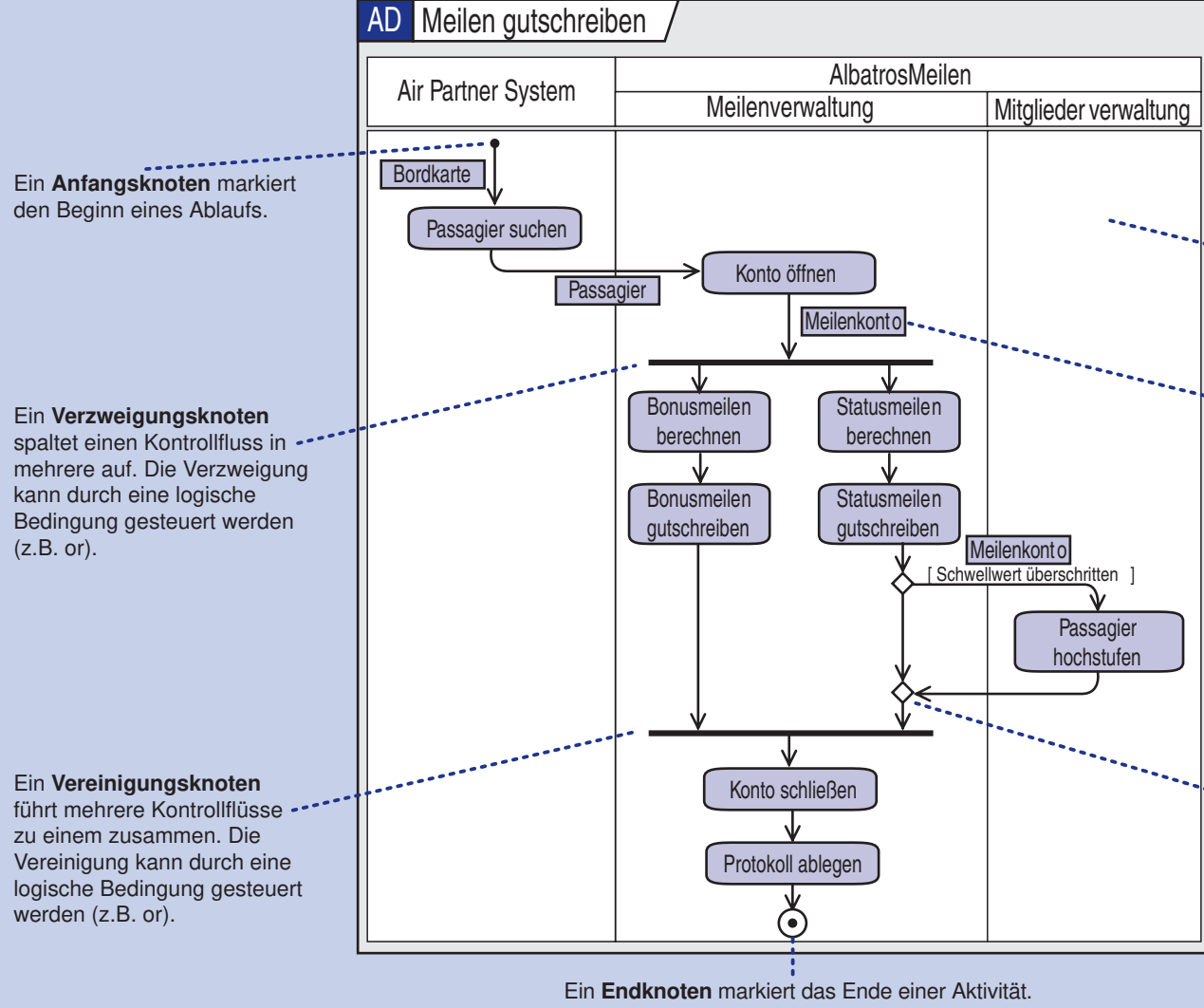


Ein **Knoten** repräsentiert einen Rechner oder eine sonstige aktive Komponente in einer Installation. Knoten können ineinander geschachtelt werden.

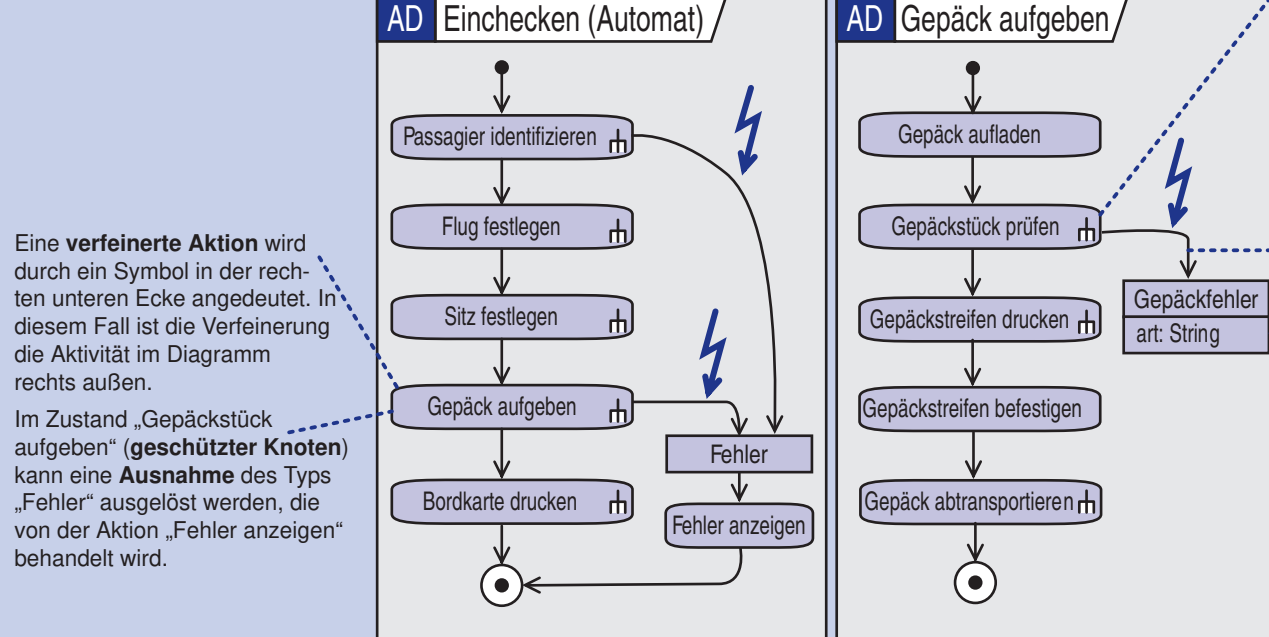


Aktivitätsdiagramme

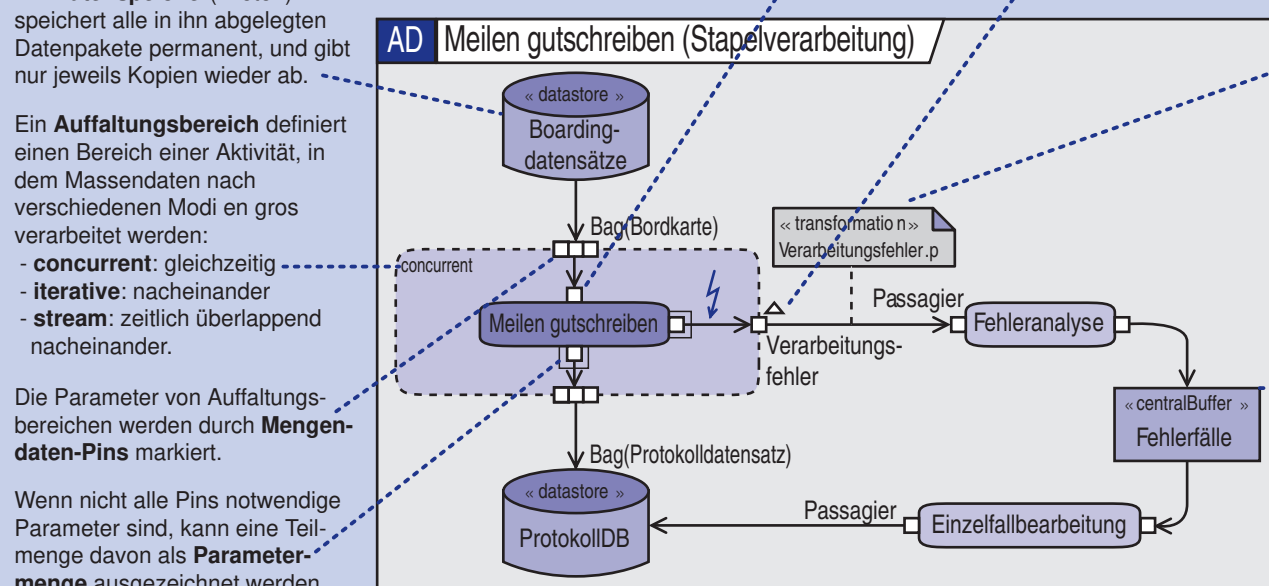
Ein **Aktivitätsdiagramm** kann zur Modellierung von Abläufen jeder Art benutzt werden, z.B. für Geschäftsprozesse, Nutzfälle oder algorithmische Abläufe.



Ein **Verzweigungsknoten** spaltet einen Kontrollfluss in mehrere auf. Die Verzweigung kann durch eine logische Bedingung gesteuert werden (z.B. or).

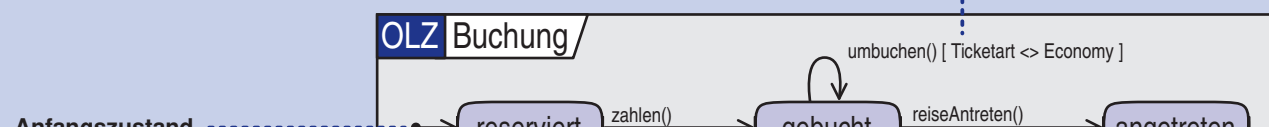


Ein **Pin** ist ein notwendiger Parameter für den Aufruf einer Aktion, bzw. ein garantiertes Resultat.

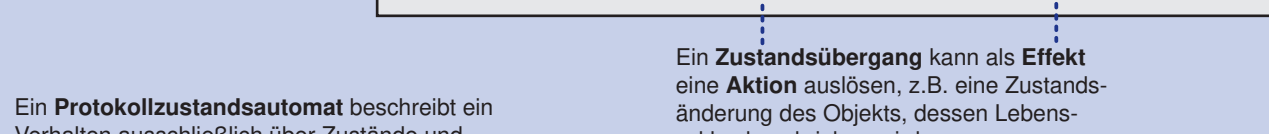


Zustandsautomaten

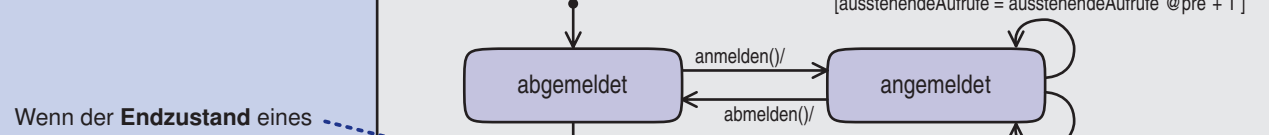
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Reisenzustandsautomat** beschreibt ein Verhalten ausschließlich über Zustände und Übergänge, aber ohne Seiteneffekte und Speicher.



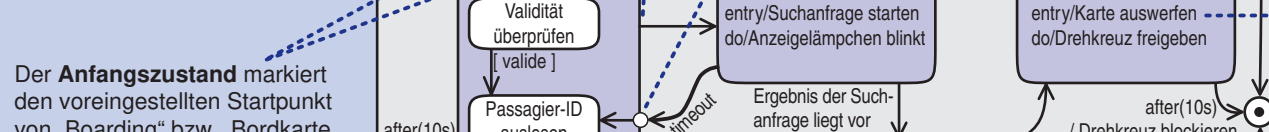
Ein **Reisenzustandsautomat** beschreibt ein Verhalten ausschließlich über Zustände und Übergänge, aber ohne Seiteneffekte und Speicher.



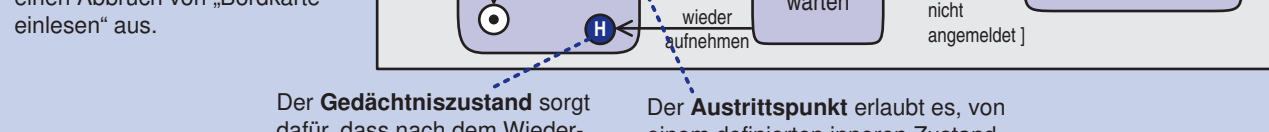
Ein **Reisenzustandsautomat** beschreibt ein Verhalten ausschließlich über Zustände und Übergänge, aber ohne Seiteneffekte und Speicher.



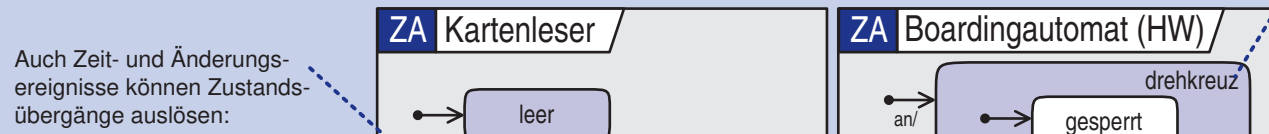
Ein **Reisenzustandsautomat** beschreibt ein Verhalten ausschließlich über Zustände und Übergänge, aber ohne Seiteneffekte und Speicher.



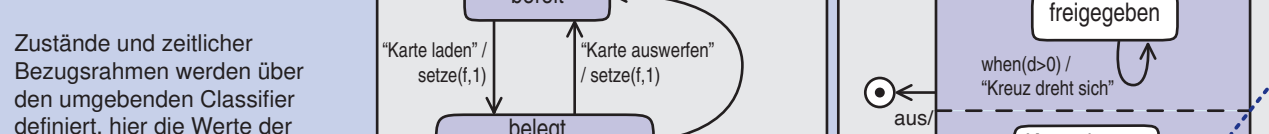
Ein **Reisenzustandsautomat** beschreibt ein Verhalten ausschließlich über Zustände und Übergänge, aber ohne Seiteneffekte und Speicher.



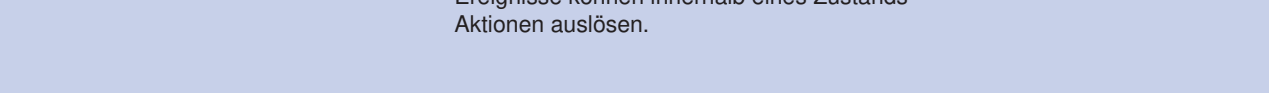
Ein **Reisenzustandsautomat** beschreibt ein Verhalten ausschließlich über Zustände und Übergänge, aber ohne Seiteneffekte und Speicher.



Ein **Reisenzustandsautomat** beschreibt ein Verhalten ausschließlich über Zustände und Übergänge, aber ohne Seiteneffekte und Speicher.

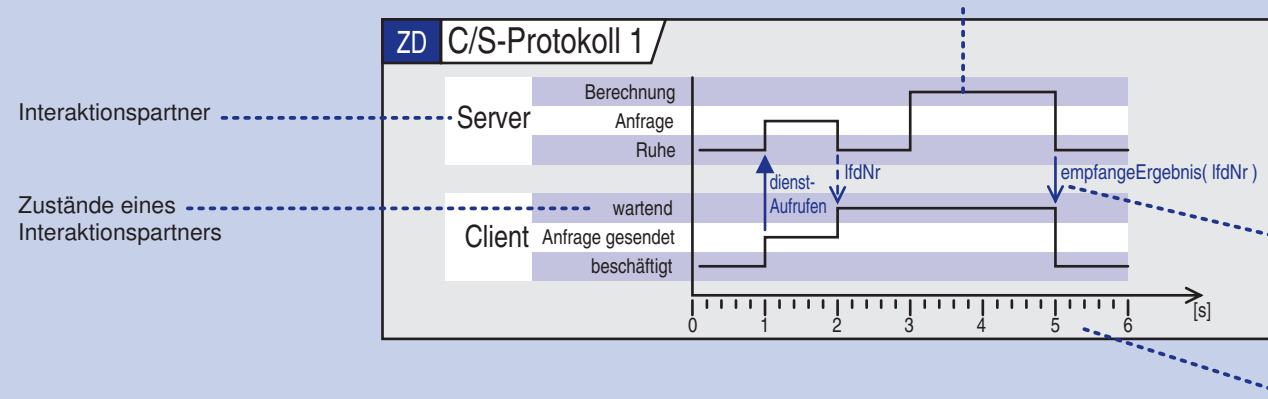


Ein **Reisenzustandsautomat** beschreibt ein Verhalten ausschließlich über Zustände und Übergänge, aber ohne Seiteneffekte und Speicher.

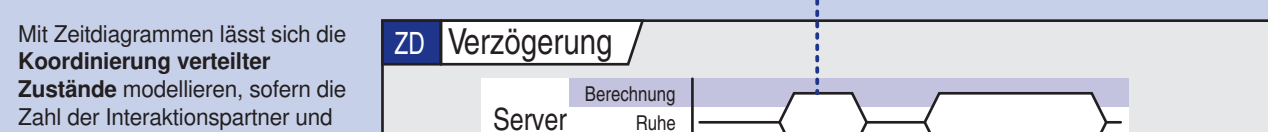


Zeitdiagramme

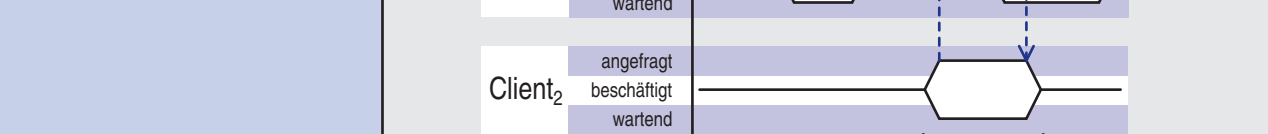
Zeitdiagramme sind eine Form von Interaktionsdiagrammen, die besonders zur Modellierung und Visualisierung komplexer zeitlicher Abläufe geeignet sind.



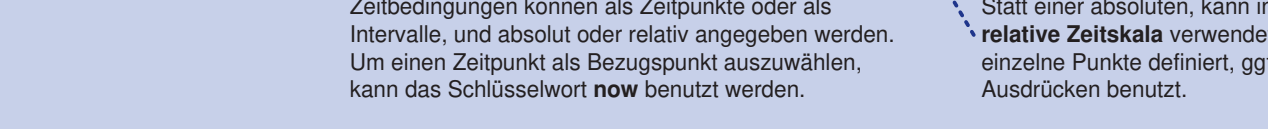
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



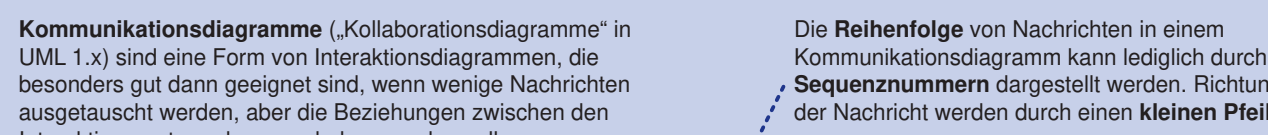
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



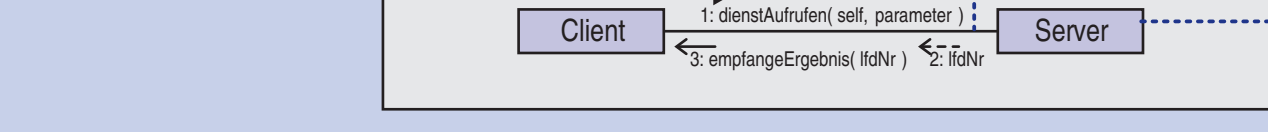
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



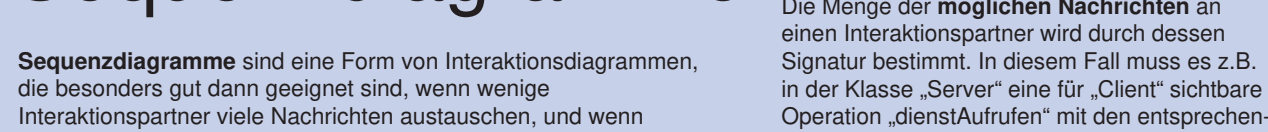
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



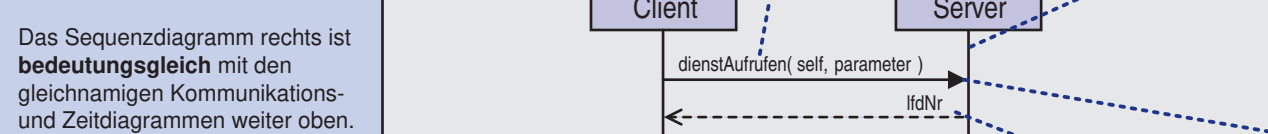
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



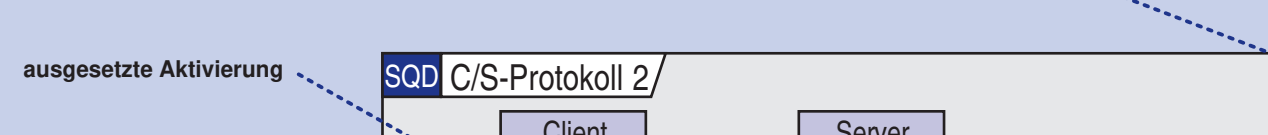
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



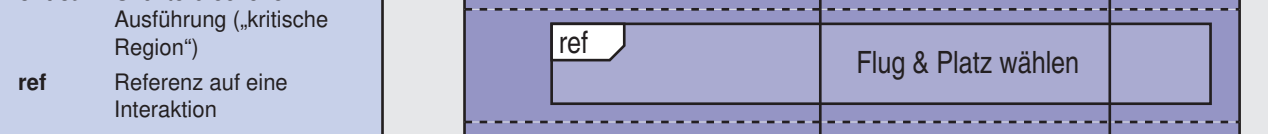
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



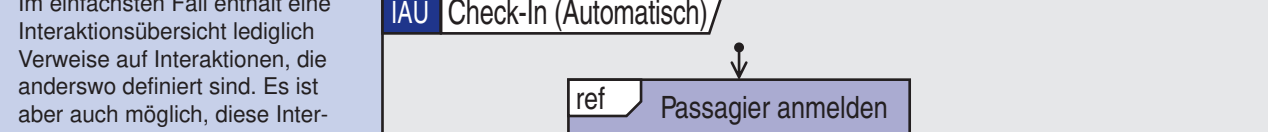
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



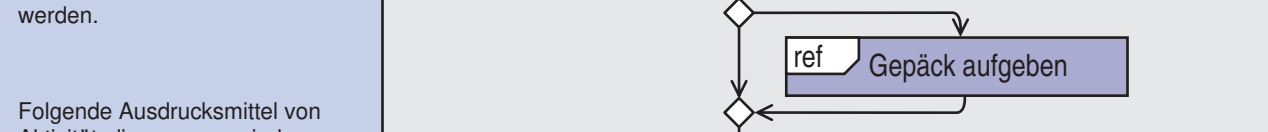
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.

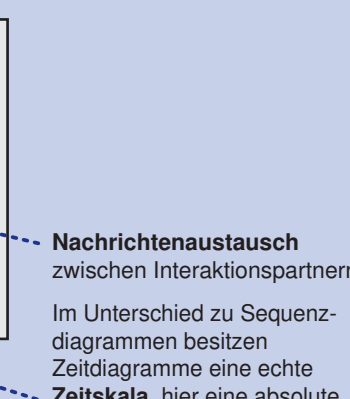


Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.

Ähnlich zu Sequenzdiagrammen besitzen auch Zeitdiagramme eine Art **Lebenslinie**, die hier aber auch den Zustand eines Interaktionspartners zu einem Zeitpunkt ausdrückt.



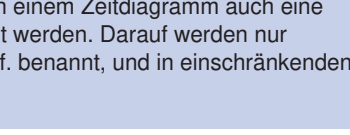
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



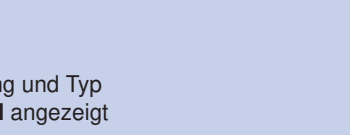
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



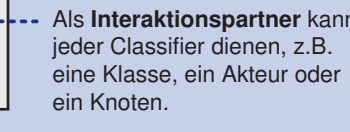
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



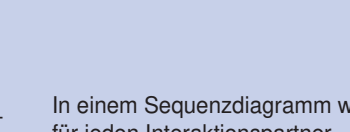
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



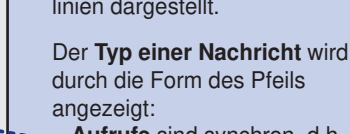
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



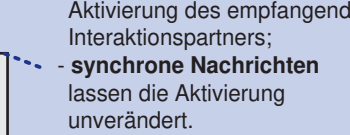
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



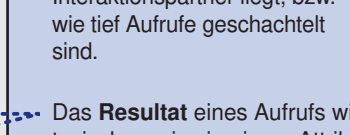
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



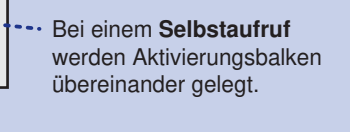
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



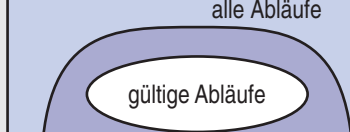
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



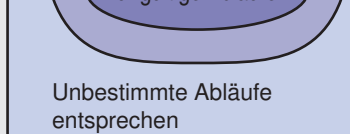
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



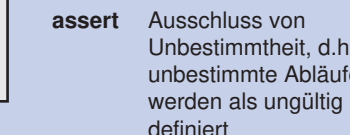
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



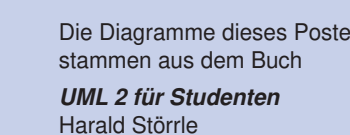
Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.



Ein **Objektzustandszyklus** beschreibt das Verhalten aller Objekte einer Klasse, von der Instanzierung zur Terminierung. Der Zustand eines Objekts besteht aus den Werten aller Attribute des Objekts.