eBook „UML2 glasklar“

Skript

Diagrammsprachen der UML 2

7 | 11 | 12 | 15 | Rahmenkopf (16)

Diagramme der UML 2 (S. 7)

Strukturdiagramme:	Klassendiagramm, Paketdiagramm, Komponentendiagramm
Verhaltensdiagramme:	Use-Case-Diagramm, Aktivitätsdiagramm, Zustandsautomat Interaktionsdiagramme: Sequenzdiagramm

Diagramme der UML 2 und ihre Anwendung (S. 11)

Klassendiagramm

- Statische Struktur
- Relevante Strukturzusammenhänge und Datentypen
- Brücke zu dynamischen Diagrammen

Paketdiagramm

Organisiert Systemmodell; logische Zusammenfassung

Komponentendiagramm

Organisation technischer Systemkomponenten

Use-Case-Diagramm

- Außensicht auf das System
- Kontextabgrenzung
- Hohes Abstraktionsniveau

Aktivitätsdiagramm

Visualisierung von Abläufen

Zustandsautomat

Zustandsmodell

Sequenzdiagramm

Ablauf des Informationsaustausches zwischen Kommunikationspartnern

Grundelemente der UML

Modell (21) | Kommentar (22) | Datentyp (26) | Aufzählungstyp (27, 29)

Modell (S. 21)

- *A model captures a view of a system. It is an abstraction of the system, with a certain purpose*
- Notation: Paketsymbol mit Schlüsselwort «model»
- Versammelt eine hierarchisch strukturierte Menge von Elementen
- Menge von Elementen beschreibt auf einer abstrakten Ebene das System

Kommentar (S. 22)

- *A comment is a textual annotation that can be attached to a set of elements*
- Notation: Rechteck mit umgeknickter rechter oberer Ecke; durch eine gestrichelte Linie mit dem annotierten Element verbunden
- Kommentar beeinflusst die Semantik des Modells nicht
- Kommentar darf an jedem beliebigen Modellelement angebracht werden

Datentyp (S. 26)

- *A data type is a type whose instances are identified only by their value*
  A data type may contain attributes to support the modeling of structured data types
- Notation: Rechtecksymbol, Schlüsselwort «datatype»
- Wertebereiche einzugrenzen
- Aufzählungstyp (Enumeration)

Aufzählungstyp (S. 27, 29)

- *An enumeration is a data type whose values are enumerated in the model as enumeration literals. An enumeration literal is a user-defined data value for an enumeration*
- Notation: wie Datentypen, Schlüsselwort «enumeration»
- Aufzählungstypen sind spezielle Datentypen mit einer endlichen Menge benutzerdefinierter, diskreter Werte

Analyse mit der UML

Hauptaufgaben der Systemanalyse (45)

Hauptaufgaben der Systemanalyse (S. 45)

- Definieren der Systemgrenzen und Schnittstellen
- Beschreiben der Funktionalitäten
- Modellieren

Die UML in der Realisierung

Hauptaufgaben der Systemarchitektur (78)

Hauptaufgaben der Systemarchitektur (S. 78)

- Zerlegen des Systems
- Verteilen von Verantwortlichkeiten
- Beschreiben der Schnittstellen

Verhaltensmodellierung

Verhaltensdiagramme der UML 2 (240)

Verhaltensdiagramme der UML 2 (S. 240)

- Use-Case-Diagramm
- Aktivitätsdiagramm
- Zustandsautomat
- Sequenzdiagramm

Use-Case-Diagramm

Grundlagen (241,242) | Use-Case (246,247) | System (249,250) | Akteur (251,252) | «include»-Beziehung (256) | «extend»-Beziehung (258) | Vergleich «include»-/«extend»-Beziehung (262)

Grundlagen (S. 241, 242)

- Beginn jeder Systementwicklung
- Use-Case-Diagramm zeigt das externe Verhalten eines Systems aus der Sicht der Nutzer
- Use-Case kapselt geschlossene Sammlung von Aktionen
- Akteur liefert oder empfängt Signale bzw. Daten zum oder vom System
- Use-Case-Ablauf ist abgeschlossener Vorgang mit einem Auslöser und einem Ergebnis
- Akteur initiiert Use-Case

Use-Case (S. 246, 247)

- *A use case is the specification of a set of actions performed by a system, which yields an observable result that is, typically, of value for one or more actors or other stakeholders of the system*
- Notation: Ellipse, Name inner- oder unterhalb
- Menge von Aktionen, die, schrittweise ausgeführt, ein spezielles Verhalten formen

System (S. 249, 250)

- *Subject: Extending a classifier with the capability to own use cases*
- Notation: rechteckiger Kasten, Kanten sind Systemgrenzen
- Einheit, die Use-Cases realisiert und anbietet

Akteur (S. 251, 252)

- *An actor specifies a role played by a user or any other system that interacts with the subject.*
- Notation: Strichmännchen,Namen oberhalb oder unterhalb
- Interagiert mit System, steht immer außerhalb
- Rolle
- In einem Use-Case-Diagramm interagiert ein Akteur mit einem Use-Case, indem er dessen Ausführung anstößt oder an der Ausführung aktiv oder passiv beteiligt ist. Zwischen dem Use-Case und dem Akteur findet ein wechselseitiger Austausch von Signalen und Daten statt (binäre Assoziation)

«include»-Beziehung (S. 256)

- *An include relationship defines that a use case contains the behavior defined in another use case*
- Notation: unterbrochene gerichtete Kante mit dem Schlüsselwort «include»
- Die «include»-Beziehung visualisiert, dass ein Use-Case A das Verhalten eines anderen Use-Case B importiert

«extend»-Beziehung (S. 258)

- *Extend: A relationship from an extending use case to an extended use case that specifies how and when the behavior defined in the extending use case can be inserted into the behavior defined in the extended use case*

  Extension Point: An extension point identifies a point in the behavior of a use case where that behavior can be extended by the behavior of some other (extending) use case, as specified by an extend relationship
- Notation: unterbrochene gerichtete Kante mit der Bezeichnung «extend» vom erweiternden Use-Case zum erweiterten Use-Case
- Die «extend»-Beziehung zeigt an, dass das Verhalten eines Use-Case (A) durch einen anderen Use-Case (B) erweitert werden kann

Vergleich «include»-/«extend»-Beziehung (262)

- Notation

- Bedeutung

  «include»-Beziehung: Ablauf von A schließt immer Ablauf von B ein
  «extend»-Beziehung: Ablauf von A kann, muss aber nicht durch Ablauf von B erweitert werden

- Nutzung der Beziehung

  «include»-Beziehung: Ablauf von B kann in verschiedenen Use-Cases genutzt werden
  «extend»-Beziehung: A besitzt neben Normalverhalten auslagerbare Sonderfälle

- Bedeutung für Modellierung

  «include»-Beziehung: A ist meist unvollständig und wird erst durch Inklusion B zu einem vollständigen Ablauf
  «extend»-Beziehung: A ist vollständig und kann durch B optional erweitert werden

- Abhängigkeiten

  «include»-Beziehung: A muss B bei der Modellierung berücksichtigen
  B wird unabhängig von A modelliert, um die Nutzung durch weitere Use-Cases sicherzustellen (Wiederverwendbarkeit),
  B muss in sich nicht vollständig sein ("B weiß nicht, durch wen er inkludiert wird")
  «extend»-Beziehung: A muss durch Angabe von Erweiterungspunkten auf die Erweiterung durch B vorbereitet werden
  B wird in sich vollständig und unabhängig von A modelliert ("B weiß nicht, wen er erweitert")

Aktivitätsdiagramm

Grundlagen (263,264) | Token-Konzept (265,267) | Aktion (274) | Aktivität (278) | Kanten (287, 288) | Startknoten (292) | Endknoten (293) | Verzweigungsknoten (296) | Synchronisations- und Parallelisierungsknoten (299,300) | Aktivitätsbereich (308)

Grundlagen (S. 263, 264)

- Modellierung von Abläufen
- Darstellung von Aktivitäten mit einem nichttrivialen Charakter
- Aktivität: Menge von potenziellen Abläufen
- Elemente: Aktivitäten, Aktionen, Objektknoten, Kontrollelemente und Kanten

Token-Konzept (S. 265, 267)

- logisches Konzept
- Token: Marke, Staffelstab
- Wanderung eines Tokens durch eine Aktivität repräsentiert die Abarbeitung
- Aktion wird von einem Token ausgelöst
- Verzweigung und Vereinigung ► Token-Routing
- Parallelabarbeitung ► Token-Vervielfältigung
- Synchronisation ► Token-Verschmelzung

Aktion (S. 274)

- *An action is a named element that is the fundamental unit of executable functionality. The execution of an action represents some transformation or processing in the modeled system, be it a computer system or otherwise*
- Notation: Rechteck mit abgerundeten Ecken
- Aufruf eines Verhaltens oder die Bearbeitung von Daten, Einzelschritt
- Aktion über Kanten mit anderen Elementen verbunden

Aktivität (S. 278)

- *An activity is the specification of parameterized behavior as the coordinated sequencing of subordinate units whose individual elements are actions*
- Notation: Rechteck mit abgerundeten Ecken
- Gesamte Einheit in einem Aktivitätsmodell

Kanten (S. 287, 288)

- *An activity edge is an abstract class for directed connections between two activity nodes*
- Übergänge zwischen zwei Knoten, immer gerichtet
- Arten: Kontrollfluss und Objektfluss
- Kontrollfluss: Kante zwischen zwei Aktionen oder zwischen einer Aktion und einem Kontrollelement
- Objektfluss: mindestens ein Objektknoten

Startknoten (S. 292)

- *An initial node is a control node at which flow starts when the activity is invoked*
- Notation: Ausgefüllter schwarzer Punkt
- Startpunkt eines Ablaufs

Endknoten (S. 293, 294)

- *A flow final node is a final node that terminates a flow*
  An activity final node is a final node that stops all flows in an activity
- Arten: Endknoten für Aktivitäten (beendet die gesamte Aktivität)
  Endknoten für Kontrollflüsse (beendet nur einen einzelnen Ablauf)
- Notation: Endknoten für Aktivitäten (schwarzer Punkt mit umschließenden Ring)
  Endknoten für Kontrollflüsse (Kreuz mit umschließenden Ring)

Verzweigungsknoten (S. 296)

- Verzweigungsknoten (decision node) spaltet eine Kante in mehrere Alternativen auf
- Token passiert (nur) eine ausgehende Kante

Synchronisations-und Parallelisierungsknoten (S. 299, 300)

- *A fork node is a control node that splits a flow into multiple concurrent flows*
  A join node is a control node that synchronizes multiple flows
- Notation: Schwarze Balken
- Parallelisierungsknoten
  - Eingehende Ablauf in mehrere parallele Abläufe aufgeteilt
  - Token wird dupliziert und an jeder ausgehenden Kante angeboten (Token-Vervielfältigung)
- Synchronisationsknoten
  - Führt eingehende Abläufe zu einem gemeinsamen Ablauf zusammen
  - Kontroll-Token werden am Synchronisationsknoten zu einem Ausgangskontroll-Token verschmolzen (Token-Verschmelzung)

Aktivitätsbereich (S. 308)

- *An activity partition is a kind of activity group for identifying actions that have some characteristic in common*
- Aktivität in Bereiche mit gemeinsamen Eigenschaften unterteilen

Paketdiagramm

Notationselemente (175) | Paket-Import (176,177) | Paket-Merge (179)

Notationselemente (S. 175)

- *A package is used to group elements, and provides a namespace for the grouped elements*
- Notation: Rechteck, Bezeichner auf der Lasche oben links
- Paket fasst mehrere paketierbare Elemente zu einer größeren Einheit zusammen
- Namensraum

Paket-Import (S. 176, 177)

- *A package import is a relationship that allows the use of unqualified names to refer to package members from other namespaces*
- Notation: gestrichelte Linie mit Pfeil, Schlüsselwort «import» oberhalb des Pfeils
- Gerichtete Beziehung zwischen zwei Paketen

Paket-Merge (S. 179)

- *A package merge defines how the contents of one package is extended by the contents of another package*
- Notation: Abhängigkeitsnotation, Stereotyp «merge»
- Erweiterung des Paket-Imports
- Neue spezialisierte Classifier werden implizit angelegt

Strukturdiagramme

Strukturdiagramme der UML 2 (105)

Klassendiagramm

Modellieren von Klassen (108) | Klassen und Objekte (110) | Klasse (115-117) | Attribut (118-120) | Operation (123,124) | Schnittstelle (129,130) | Generalisierung (135,138) | Generalisierungsmenge (140) | Assoziation (142,144) | Aggregation und Komposition (152,153), Assoziationsklasse (157, 158) | Abhängigkeitsbeziehung (159,160) | Verwendungsbeziehung (161) | Abstraktionsbeziehung (162) | Realisierungsbeziehung (164)

Modellieren von Klassen (S. 108)

- Objekte ► Abstraktion zu Klassen
- Klasse: Sammlung von Exemplaren, die über gemeinsame Eigenschaften, Einschränkungen und Semantik verfügen

Klassen und Objekte (S. 110)

Klasse ► Interpretation als Typ, Objekte sind Ausprägungen

Klasse (S. 115-117)

- *A class describes a set of objects that share the same specifications of features, constraints, and semantics*
- Notation: Rechteck mit durchgezogener Linie
- Menge von Objekten mit gemeinsamer Semantik, gemeinsamen Eigenschaften und gemeinsamem Verhalten
- Eigenschaften werden durch Attribute und das Verhalten durch Operationen abgebildet
- **Abstrakte Klasse**
  - Unvollständig und nicht instanziierbar, Strukturierungsmerkmal
  - Notation: wie Klasse, Name kursiv oder Schlüsselwort «abstract»

Attribut (S. 118-120)

- *A property is a structural feature. A property related to a classifier by owned Attribute represents an attribute. It relates an instance of the class to a value or collection of values of the type of the attribute*
- Notation: 2. Abschnitt im Rechteck des Klassensymbols
- Repräsentieren strukturelle Eigenschaften, Datengerüst
- Sichtbarkeit: Definition von Sichtbarkeits- und Zugriffsrestriktionen
  - public(+) : uneingeschränkter Zugriff
  - private (-) : nur Instanzen der Klasse
  - protected(#) : nur Instanzen der Klasse, Instanzen abgeleiteter Klasse
- Typ: Datentypen, eigendefinierte Typen durch Klassen
- Multiplizität: Unter- und Obergrenze Anzahl der ablegbare Instanzen
  - 0..1 : optionales Attribut, höchstens ein Wert
  - 1..1, altern. 1 : zwingendes Attribut, genau ein Wert
  - 0..*, altern. * : optional beliebig, beliebig viele Werte, auch null
  - 1..* : beliebig viele, mindestens ein Wert
  - n..m : fixiert, mindestens n und höchstens m Werte
- Vorgabewert: Angabe eines Wertes der bei Erzeugung Objekt automatisch gesetzt wird

Operation (S. 123, 124)

- *An operation is a behavioral feature of a classifier that specifies the name, type, parameters, and constraints for invoking an associated behavior*
- Notation: 3. Abschnitt im Rechteck des Klassensymbols
- Verhalten von Objekten
- Zustandsänderung herbeiführen
- Signatur: Name, Über-und Rückgabeparameter
- Methode: Implementierung
- Botschaften: Nach einer gängigen abstrakten Sichtweise kommunizieren die Objekte eines Systems miteinander, indem sie Botschaften austauschen, die dann zum Ablauf des in einer Methode abgelegten Codes führen

Schnittstelle (S. 129, 130)

- *An interface is a kind of classifier that represents a declaration of a set of coherent public features and obligations. An interface specifies a contract; any instance of a classifier that realizes the interface must fulfill that contract*
- An interface realization is a specialized realization relationship between a classifier and an interface. This relationship signifies that the realizing classifier conforms to the contract specified by the Interface
- Notation: 1. nicht ausgefülltes Kreissymbol (Implementierung), Lollipop-Darstellung
  2. Pfeil mit dem leeren Dreieck am Ende zeigt auf Implementierung
- Menge von öffentlichen Operationen, Merkmalen und "Verpflichtungen"

Generalisierung (S. 135, 138)

- *A generalization is a taxonomic relationship between a more general classifier and a more specific classifier. Each instance of the specific classifier is also an indirect instance of the general classifier. Thus, the specific classifier inherits the features of the more general classifier*
- Notation:
  - nicht ausgefüllte dreieckige Pfeilspitze
  - zeigt vom spezialisierten Classifier hin zum allgemeinen Classifier
- Zentrales Konzept objektorientierter Modellierung
- Umkehrung der Generalisierung ist die Spezialisierung
- Überschreiben: Unterklasse implementiert eine geerbte Operation der Oberklasse neu
- Erhalt der Substitutionsfähigkeit beim Überschreiben

Generalisierungsmenge (S. 140)

- *A generalization set is a packageable element whose instances define collections of subsets of generalization relationships*
- Notation: Name, Eigenschaften in geschweiften Klammern
- Fasst eine Menge von Subtypen einer Generalisierung nach Schema zusammen (Abstraktionsschritt der Generalisierung)

Assoziation (S. 142, 144)

- *An association describes a set of tuples whose values refer to typed instances. An instance of an association is called a link*
- Notation: Darstellung von Beziehungen zwischen Klassen
  durchgezogene Linie, binäre Assoziation
- Menge gleichartiger Beziehungen zwischen Klassen

Aggregation und Komposition (S. 152, 153)

- **Aggregation**
  - Notation: nicht-ausgefülltes Diamantsymbol auf der Seite des Aggregat
  - Teile-Ganzes-Beziehung
  - Semantik "besteht aus"
  - lose Teile-Ganzes-Beziehung
- **Komposition**
  - Notation: ausgefülltes Diamantsymbol auf der Seite des Aggregat
  - Physische Inklusion der Teile im Ganzen
  - Einschränkung, dass ein Teil zu einem Zeitpunkt höchstens genau einem Ganzen zugeordnet sein darf
  - Lebensdauer des Teils hängt von der des Ganzen
  - enge Teile-Ganzes-Beziehung

Assoziationsklasse (S. 157, 158)

- *A model element that has both association and class properties. An association class can be seen as an association that also has class properties, or as a class that also has association properties. It not only connects a set of classifiers but also defines a set of features that belong to the relationship itself and not to any of the classifiers*
- Notation: Klasse, über unterbrochene Linie mit einer Assoziation verbunden
- Eigenschaften Klasse und Assoziation vereint
- Eigenschaften, die keinem der zur Assoziation beitragenden Classifier zugeordnet werden

Abhängigkeitsbeziehung (S. 159, 160)

- *A dependency is a relationship that signifies that a single or a set of model elements requires other model elements for their specification or implementation. This means that the complete semantics of the depending elements is either semantically or structurally dependent on the definition of the supplier element(s)*
- Notation: gestrichelter Pfeil, Pfeil zeigt vom abhängigen zum unabhängigen Element
- Modellierung von Abhängigkeiten zwischen verschiedenen Modellelementen

Verwendungsbeziehung (S. 161)

- *A usage is a relationship in which one element requires another element (or set of elements) for its full implementation or operation. In the metamodel, a usage is a dependency in which the client requires the presence of the supplier*
- Notation: Abhängigkeitsbeziehung, gestrichelte Linie mit Pfeil, Schlüsselwort «use»
  Pfeil zeigt vom "unvollständigen" Element auf das benötigte Element
- Verbindet ein oder mehrere abhängige Modellelemente mit einem oder mehreren Elementen, die für technische Realisierung abhängiger Modellelemente erforderlich sind

Abstraktionsbeziehung (S. 162)

- *An abstraction is a relationship that relates two elements or sets of elements that represent the same concept at different levels of abstraction or from different viewpoints*
- Notation: Abhängigkeitsbez., gestrichelte Linie mit Pfeil, Schlüsselwort «abstraction»
- Abstraktionsbeziehung (abstraction) verbindet zwei oder auch mehr Elemente, die sich auf unterschiedlichen Abstraktionsebenen befinden oder unterschiedliche Sichten darstellen

Realisierungsbeziehung (S. 164)

- *Specialized abstraction relationship between two sets of model elements, one representing a specification (the supplier) and the other representing an implementation of the latter*
- Notation: spezielle Form der Abstraktionsbeziehung
  geschlossenes Dreieck am Ende der gestrichelten Linie
  Supplier-Element (am Pfeil) als Spezifikation
  Client-Elemente als Realisierung der Spezifikation gegenüber
- Spezielle Abstraktionsbeziehung zwischen einzelnen Elementen

Zustandsdiagramm

Notationselemente (330) | Einfacher Zustand (337,338) | Transition (340-342) | Startzustand (346,347) | Endzustand (348) | Pseudozustände (349) | Kreuzung (351,352) | Entscheidung (353,354) | Gabelung und Vereinigung (363,364)

Notationselemente (S. 330)

- Erweiterung der endlichen Automaten
- Annahmen:
  - System befindet sich zu einem bestimmten Zeitpunkt genau in einem Zustand
  - Übergang (Transition) von einem Zustand in den nächsten ohne zeitliche Verzögerung
- Notationselemente:
  Zustände, Transitionen, Zustandsautomat, Regionen, Start- und Endzustände
- Arten von Zuständen:
  Einfache Zustände, Zusammengesetzte Zustände

Einfacher Zustand (S. 337, 338)

- *A state models a situation during which some (usually implicit) invariant condition holds*
- Notation: Rechteck mit abgerundeten Ecken
  Schlüsselwörter Verhalten ► entry (Eintritt), exit (Austritt), do (Zustand)
- Abbildung einer Situation, in deren Verlauf eine spezielle Bedingung gilt

Transition (S. 340-342)

- *A transition is a directed relationship between a source vertex and a target vertex. It may be part of a compound transition, which takes the state machine from one state configuration to another, representing the complete response of the state machine to an occurrence of an event of a particular type*
- Notation: durchgezogene, gerichtete und beschriftete Kante
  Trigger: Auslöser für die Transition
  Guard: Bedingung, die wahr sein muss, damit die Transition bei Erhalt des Triggers durchlaufen wird, in eckigen Klammern
  Verhalten: Verhalten beim Durchlaufen der Transition
- Übergang von einem Ausgangs- zu einem Zielzustand

Startzustand (S. 346, 347)

- *An initial pseudostate represents a default vertex that is the source for a single transition to the default state of a composite state. There can be at most one initial vertex in a region. The initial transition may have an action*
- Notation: ausgefüllter Kreis
- Startpunkt für das Betreten eines Zustandsautomaten
- **Einschränkungen**
  weder einen Trigger noch eine Bedingung, eine ausgehende Transition

Endzustand (S. 348)

- *Final state is special kind of state signifying that the enclosing region is completed. If the enclosing region is directly contained in a state machine and all other regions in the state machine also are completed, then it means that the entire state machine is completed*
- Notation: kleiner ausgefüllter Kreis, umgeben von einem unausgefüllten Kreis
- Beendigung der Abarbeitung

Pseudozustände (S. 349)

- *A pseudostate is an abstraction that encompasses different types of transient vertices in the state machine graph. Pseudostates are typically used to connect multiple transitions into more complex state transitions paths*
- Notationen: Startzustand, Entscheidung, Kreuzung, Gabelung, Vereinigung

Kreuzung (S. 351, 352)

- *Junction vertices are semantic-free vertices that are used to chain together multiple transitions. They are used to construct compound transition paths between states*
- Notation: kleiner ausgefüllter Kreis
- Transitionen ohne verbindende Zustände hintereinander schalten

Entscheidung (S. 353, 354)

- *Choice vertices which, when reached, result in the dynamic evaluation of the guards of the triggers of its outgoing transitions*
- Notation: nicht ausgefüllte Raute
- Abzweigstellen

Gabelung und Vereinigung (S. 363, 364)

- *Fork vertices serve to split an incoming transition into two or more transitions terminating on orthogonal target vertices (i.e. vertices in different regions of a composite state). The segments outgoing from a fork vertex must not have guards or triggers*
- *Join vertices serve to merge several transitions emanating from source vertices in different orthogonal regions. The transitions entering a join vertex cannot have guards or triggers*
- Notation: kurzer dicker senkrechter Strich
- **Gabelung**
  - Aufteilung eingehende Transition auf mehrere parallele Ziele
  - eine eingehende und mindestens zwei ausgehende Transitionen
- **Vereinigung**
  - Zusammenführung von - durch Gabelung - aufgeteilte Transitionen
  - eine ausgehende und mindestens zwei eingehende Transitionen

Sequenzdiagramm

Interaktion (403,404,408,419) | Lebenslinie (422) | Nachricht (428-430)

Interaktion (S. 403, 404, 408, 419)

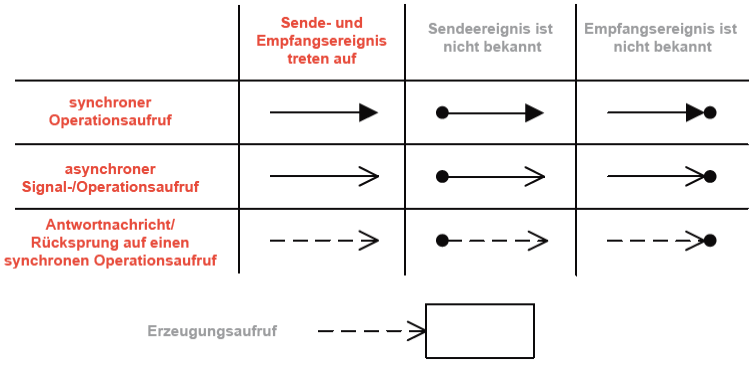
- *An interaction is a unit of behavior that focuses on the observable exchange of information between connectable elements*
- Notation: Interaktionen werden in einem Rahmen umschlossen
- Interaktion ist das Zusammenspiel von mehreren Kommunikationspartnern
- Nachrichten- und Datenaustausch
- Grundelemente ► Lebenslinien und Nachrichten
- **Grundprinzip einer Interaktion**
  Ereignismodell ► Sender, Nachricht, Empfänger
- **Dimensionen**
  - zwei Dimensionen (zwei Leserichtungen)
  - links nach rechts: Kommunikationspartner (Lebenslinien)
  - oben nach unten: Zeitachse
  - vertikal: zeitliche Abfolge Kommunikationsschritte
  - horizontal: kommunizierende Partner
- **Grundbausteine**
  - Kommunikationspartner als Lebenslinien
  - Nachrichten in Form von Pfeilen

Lebenslinie (S. 422)

- *A lifeline represents an individual participant in the interaction*
- Notation: rechteckiger unausgefüllter Kasten mit angeschlossener Linie
- Lebenslinie (lifeline) repräsentiert genau einen Teilnehmer

Nachricht (S. 428-430)

- *A message defines a particular communication between lifelines of an interaction*
- Notation:



- Informationsfluss zwischen Kommunikationspartnern, Sender zum Empfänger
- Aufruf/Rücksprung einer Operation

Kompositionsstrukturdiagramm

Kapselung durch Ports (202) | Port (211)

Port (S. 211) und Kapselung durch Ports (S. 202)

- *A port is a property of a classifier that specifies a distinct interaction point between that classifier and its environment or between the (behavior of the) classifier and its internal parts*
- Notation: kleines Quadrat an der Umrandung des Classifiers
- definierte Kommunikationsschnittstelle
- Zugangspunkt für Interaktionen zwischen den internen Strukturen und der Umgebung des Classifiers
- Durch diese Trennung ist es möglich, den Classifier entsprechend seiner durch die Ports definierten Einsatzfähigkeiten wiederzuverwenden

Komponentendiagramm

Grundlagen (216) | Notationselemente (220,221) | Artefakt (223)

Grundlagen (S. 216)

Bereitstellung abgegrenztes und über definierte Schnittstellen zugreifbares Verhalten

Notationselemente (S. 220, 221)

- *A component represents a modular part of a system that encapsulates its contents and whose manifestation is replaceable within its environment*
  A component defines its behavior in terms of provided and required interfaces
- Notation: Klassensymbol, Schlüsselwort «component»
- Modularer Systemteil, der seinen Inhalt transparent kapselt und verbirgt
- Eigenständige Anwendung
- Funktionalität wird über extern nutzbare Schnittstellen angeboten bzw. durch Nutzung von Schnittstellen, die durch andere Komponenten angeboten werden, erreicht

Artefakt (S. 223)

- An artifact is the specification of a physical piece of information that is used or produced by a software development process, or by deployment and operation of a system
- Notation: Classifier-Symbol, Schlüsselwort «artifact»
- Physische Informationseinheit, beispielsweise ausführbare Binärdateien