Software Engineering in der Praxis

Praktische Übungen

Objektorientiertes Design

Florin Pinte Marc Spisländer

Lehrstuhl für Software Engineering
Friedrich-Alexander-Universität Erlangen-Nürnberg

19. Januar 2010

Modelchecking

- CTL-Formeln
- NuSMV: Synchrone und asynchrone Automaten
- Lebendigkeit als Begriff mit Abstufungen
- NuSMV: Nichtdeterminismus
- interaktive Simulation und automatischer Beweis

Analysewerkzeuge der UML

- Anwendungsfalldiagramme
  - Bestimmung, welche Leistungen des Systems erbracht werden
  - Vorwiegend aktionsorientiert
  - Sicht auf die Systemgrenzen
- Aktivitätendiagramme
  - Kontroll- und Datenfluß kann dargestellt werden
  - Swimlane ordnet Aktionen und Kontrollflußelemente zu Systemstrukturen
- Zustandsautomaten / Statecharts
  - Zu jedem Zeitpunkt eindeutiger Systemzustand
  - Ereignisse können Übergänge auslösen: Trigger [Guard] / Effekt

Analysewerkzeuge der UML

- Anwendungsfalldiagramme
  - Bestimmung, welche Leistungen des Systems erbracht werden
  - Vorwiegend aktionsorientiert
  - Sicht auf die Systemgrenzen
- Aktivitätendiagramme
  - Kontroll- und Datenfluß kann dargestellt werden
  - Swimlane ordnet Aktionen und Kontrollflußelemente zu Systemstrukturen
- Zustandsautomaten / Statecharts
  - Zu jedem Zeitpunkt eindeutiger Systemzustand
  - Ereignisse können Übergänge auslösen: Trigger [Guard] / Effekt

Analysewerkzeuge der UML

- Anwendungsfalldiagramme
  - Bestimmung, welche Leistungen des Systems erbracht werden
  - Vorwiegend aktionsorientiert
  - Sicht auf die Systemgrenzen
- Aktivitätendiagramme
  - Kontroll- und Datenfluß kann dargestellt werden
  - Swimlane ordnet Aktionen und Kontrollflußelemente zu Systemstrukturen
- Zustandsautomaten / Statecharts
  - Zu jedem Zeitpunkt eindeutiger Systemzustand
  - Ereignisse können Übergänge auslösen: Trigger [Guard] / Effekt

Abgrenzung

- Objektorientierte Analyse als Lernprozeß
- Heute: gestalterischer Design-Prozeß
- Vorgehensmodell: meist keine streng getrennten Phasen
- zum Teil gleiche Werkzeuge (s. Aktivitätendiagramme)

Ziel des Objektorientierten Design

- Modellieren, *wie* das System die Aufgaben löst
- Nahe an der Implementierung

Abgrenzung

- Objektorientierte Analyse als Lernprozeß
- Heute: gestalterischer Design-Prozeß
- Vorgehensmodell: meist keine streng getrennten Phasen
- zum Teil gleiche Werkzeuge (s. Aktivitätendiagramme)

Ziel des Objektorientierten Design

- Modellieren, *wie* das System die Aufgaben löst
- Nahe an der Implementierung

Abgrenzung

- Objektorientierte Analyse als Lernprozeß
- Heute: gestalterischer Design-Prozeß
- Vorgehensmodell: meist keine streng getrennten Phasen
- zum Teil gleiche Werkzeuge (s. Aktivitätendiagramme)

Ziel des Objektorientierten Design

- Modellieren, *wie* das System die Aufgaben löst
- Nahe an der Implementierung

Abgrenzung

- Objektorientierte Analyse als Lernprozeß
- Heute: gestalterischer Design-Prozeß
- Vorgehensmodell: meist keine streng getrennten Phasen
- zum Teil gleiche Werkzeuge (s. Aktivitätendiagramme)

Ziel des Objektorientierten Design

- Modellieren, *wie* das System die Aufgaben löst
- Nahe an der Implementierung

Strukturmodellierung in der UML

- Klassendiagramme
- Komponentendiagramme
- Objektdiagramme
- Paketdiagramme, Kompositionsstrukturdiagramme, Verteilungsdiagramme
- Literatur: [**?**]

Strukturmodellierung in der UML

- Klassendiagramme
- Komponentendiagramme
- Objektdiagramme
- Paketdiagramme, Kompositionsstrukturdiagramme, Verteilungsdiagramme
- Literatur: [**?**]

Strukturmodellierung in der UML

- Klassendiagramme
- Komponentendiagramme
- Objektdiagramme
- Paketdiagramme, Kompositionsstrukturdiagramme, Verteilungsdiagramme
- Literatur: [**?**]

Strukturmodellierung in der UML

- Klassendiagramme
- Komponentendiagramme
- Objektdiagramme
- Paketdiagramme, Kompositionsstrukturdiagramme, Verteilungsdiagramme
- Literatur: [**?**]

Komponentendiagramme

- Komponentendiagramme stellen die Struktur eines Systems zur Laufzeit dar

Komponente

»A component represents a modular part of a system that encapsulates its contents and whose manifestation is replacable within its environment« [1]

- Schnittstellen, Ports (“Kommunikationspunkte”), Artefakte (phys. Informationseinheiten)
- . . . uvm . . . [?]

[1] http://www.omg.org/technology/documents/modeling_spec_catalog.htm#UML

Komponentendiagramme

- Komponentendiagramme stellen die Struktur eines Systems zur Laufzeit dar

Komponente

»A component represents a modular part of a system that encapsulates its contents and whose manifestation is replacable within its environment« [1]

- Schnittstellen, Ports (“Kommunikationspunkte”), Artefakte (phys. Informationseinheiten)
- . . . uvm . . . [?]

[1] http://www.omg.org/technology/documents/modeling_spec_catalog.htm#UML

Komponentendiagramme

- Komponentendiagramme stellen die Struktur eines Systems zur Laufzeit dar

Komponente

»A component represents a modular part of a system that encapsulates its contents and whose manifestation is replacable within its environment« [1]

- Schnittstellen, Ports (“Kommunikationspunkte”), Artefakte (phys. Informationseinheiten)
- . . . uvm . . . [?]

[1] http://www.omg.org/technology/documents/modeling_spec_catalog.htm#UML

Komponentendiagramme

- Komponentendiagramme stellen die Struktur eines Systems zur Laufzeit dar

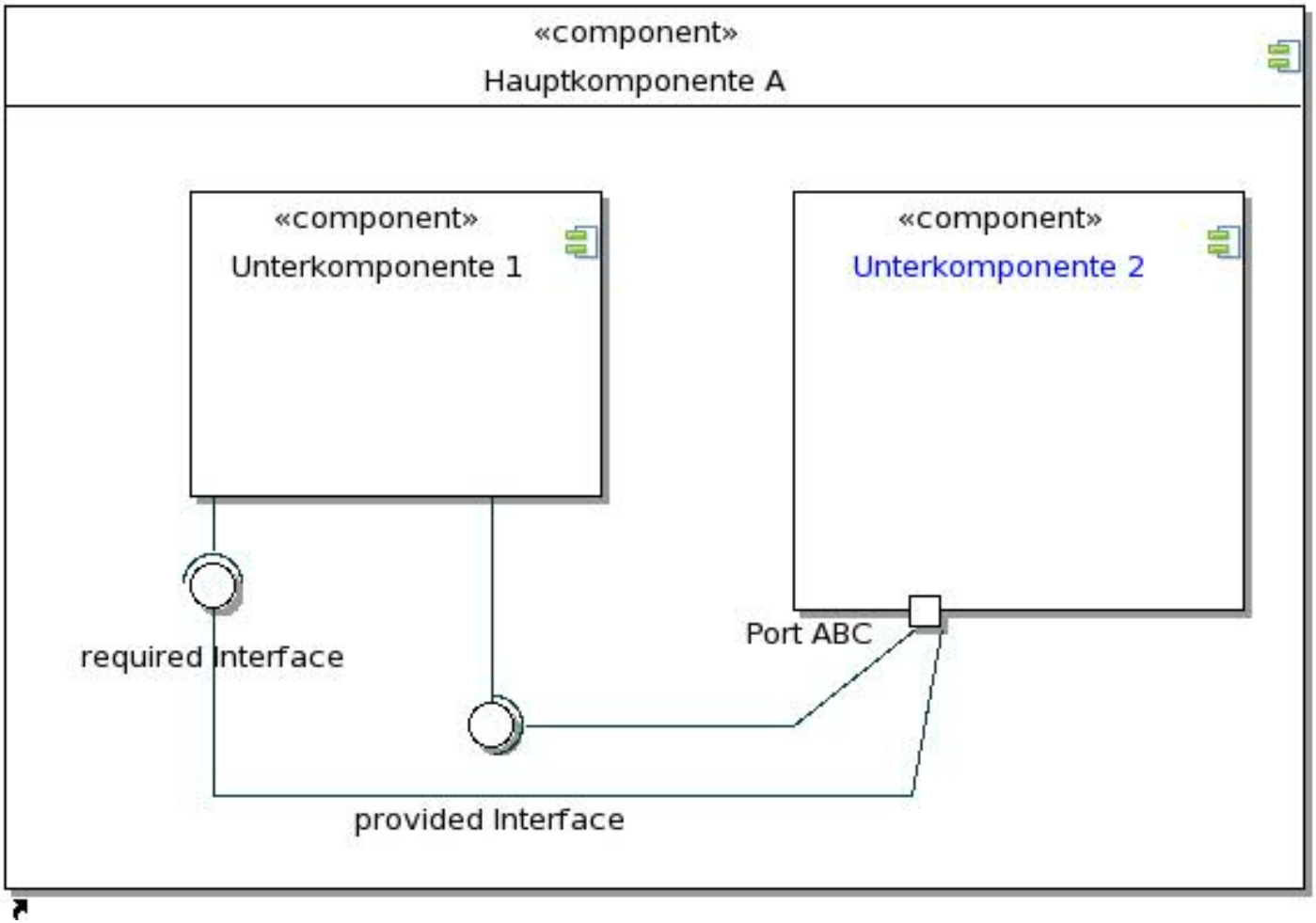
Komponente

»A component represents a modular part of a system that encapsulates its contents and whose manifestation is replacable within its environment« [1]

- Schnittstellen, Ports (“Kommunikationspunkte”), Artefakte (phys. Informationseinheiten)
- . . . uvm . . . [**?**]

[1] `http://www.omg.org/technology/documents/modeling_spec_catalog.htm#UML`

Beispiel


«component»
Hauptkomponente A
«component»
Unterkomponente 1
«component»
Unterkomponente 2
required Interface
Port ABC
provided Interface

Klassendiagramme

- Klassen: Kapselung von Attributen und Operationen
- Schnittstellen: meist nur Operationen (UML2: auch Attribute)
- Beziehungen
  - Generalisierung, Vererbung
  - Realisierung, Implementierung
  - Kompositionen
  - Aggregation
  - Assoziation
- Abhängigkeit, Verwendung, Abstraktion, Substitution, Informationsfluss, . . .

Klassendiagramme

- Klassen: Kapselung von Attributen und Operationen
- Schnittstellen: meist nur Operationen (UML2: auch Attribute)
- Beziehungen
  - Generalisierung, Vererbung
  - Realisierung, Implementierung
  - Kompositionen
  - Aggregation
  - Assoziation
- Abhängigkeit, Verwendung, Abstraktion, Substitution, Informationsfluss, . . .

Klassendiagramme

- Klassen: Kapselung von Attributen und Operationen
- Schnittstellen: meist nur Operationen (UML2: auch Attribute)
- Beziehungen
  - Generalisierung, Vererbung
  - Realisierung, Implementierung
  - Kompositionen
  - Aggregation
  - Assoziation
- Abhängigkeit, Verwendung, Abstraktion, Substitution, Informationsfluss, . . .

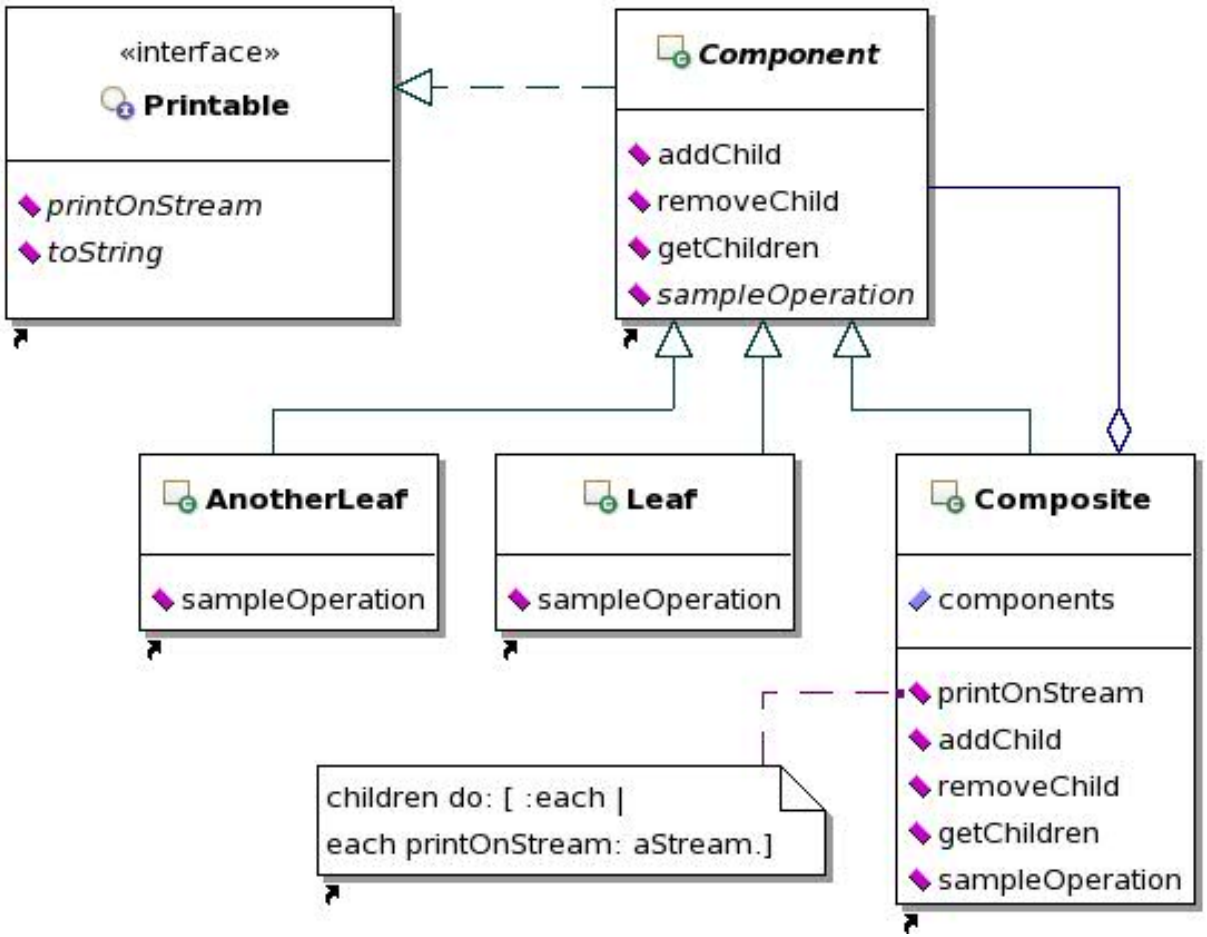
Klassendiagramme

- Klassen: Kapselung von Attributen und Operationen
- Schnittstellen: meist nur Operationen (UML2: auch Attribute)
- Beziehungen
  - Generalisierung, Vererbung
  - Realisierung, Implementierung
  - Kompositionen
  - Aggregation
  - Assoziation
- Abhängigkeit, Verwendung, Abstraktion, Substitution, Informationsfluss, . . .

Klassendiagramme

- Klassen: Kapselung von Attributen und Operationen
- Schnittstellen: meist nur Operationen (UML2: auch Attribute)
- Beziehungen
  - Generalisierung, Vererbung
  - Realisierung, Implementierung
  - Kompositionen
  - Aggregation
  - Assoziation
- Abhängigkeit, Verwendung, Abstraktion, Substitution, Informationsfluss, . . .

Klassendiagramme



Verhaltensmodellierung in der UML

- Aktivitätsmodell
  - »Was geschieht in welcher Reihenfolge?«
  - z.B. Aktivitätendiagramm
- Interaktionsmodell
  - »Wann ruft wer wen wie auf?«
  - z.B. Sequenzdiagramm oder Timingdiagramm
- Zustandsmodell
  - »Wie reagiert ein Objekt auf Ereignisse?«
  - Zustandsautomaten
- . . . uvm . . . [**?**]

Verhaltensmodellierung in der UML

- Aktivitätsmodell
  - »Was geschieht in welcher Reihenfolge?«
  - z.B. Aktivitätendiagramm
- Interaktionsmodell
  - »Wann ruft wer wen wie auf?«
  - z.B. Sequenzdiagramm oder Timingdiagramm
- Zustandsmodell
  - »Wie reagiert ein Objekt auf Ereignisse?«
  - Zustandsautomaten
- . . . uvm . . . [**?**]

Verhaltensmodellierung in der UML

- Aktivitätsmodell
  - »Was geschieht in welcher Reihenfolge?«
  - z.B. Aktivitätendiagramm
- Interaktionsmodell
  - »Wann ruft wer wen wie auf?«
  - z.B. Sequenzdiagramm oder Timingdiagramm
- Zustandsmodell
  - »Wie reagiert ein Objekt auf Ereignisse?«
  - Zustandsautomaten
- . . . uvm . . . [**?**]

Verhaltensmodellierung in der UML

- Aktivitätsmodell
  - »Was geschieht in welcher Reihenfolge?«
  - z.B. Aktivitätendiagramm
- Interaktionsmodell
  - »Wann ruft wer wen wie auf?«
  - z.B. Sequenzdiagramm oder Timingdiagramm
- Zustandsmodell
  - »Wie reagiert ein Objekt auf Ereignisse?«
  - Zustandsautomaten
- . . . uvm . . . [**?**]

UML-Zustandsautomaten

- Zustände: Name, Verhalten $(3\times)$
- Übergänge: Trigger [Guards] / Verhalten
- Entscheidungen
- Pseudozustände, Historien, Hierarchien, Regionen . . .
- Erneut: [**?**]

UML-Zustandsautomaten

- Zustände: Name, Verhalten $(3\times)$
- Übergänge: Trigger [Guards] / Verhalten
- Entscheidungen
- Pseudozustände, Historien, Hierarchien, Regionen . . .
- Erneut: [**?**]

UML-Zustandsautomaten

- Zustände: Name, Verhalten $(3\times)$
- Übergänge: Trigger [Guards] / Verhalten
- Entscheidungen
- Pseudozustände, Historien, Hierarchien, Regionen . . .
- Erneut: [**?**]

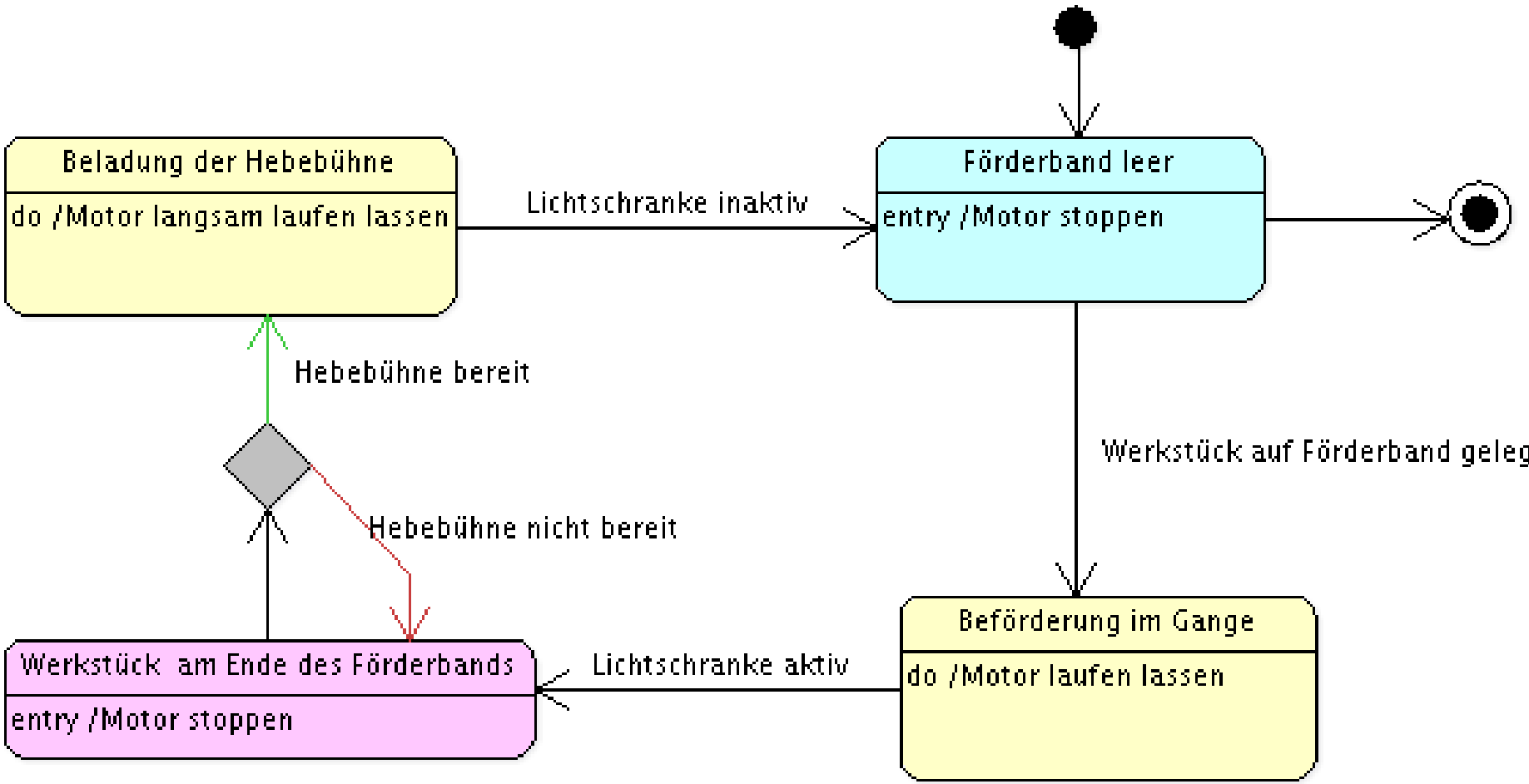
UML-Zustandsautomaten

- Zustände: Name, Verhalten $(3\times)$
- Übergänge: Trigger [Guards] / Verhalten
- Entscheidungen
- Pseudozustände, Historien, Hierarchien, Regionen . . .
- Erneut: [**?**]

UML-Zustandsautomaten

- Zustände: Name, Verhalten $(3\times)$
- Übergänge: Trigger [Guards] / Verhalten
- Entscheidungen
- Pseudozustände, Historien, Hierarchien, Regionen . . .
- Erneut: [**?**]

Zustandsautomat



Sequenzendiagramm

- Stellt ein oder mehrere Szenarien als *eine Interaktion* dar
- Objekte, Lebenslinien, Nachrichten (Methoden)
- darüber hinaus: Kontrollfluß und Schachtelung
  - optionale Ausführung
  - alternative Ausführungen
  - Referenzen auf andere Interaktionsdiagramme
  - Sprungmarken

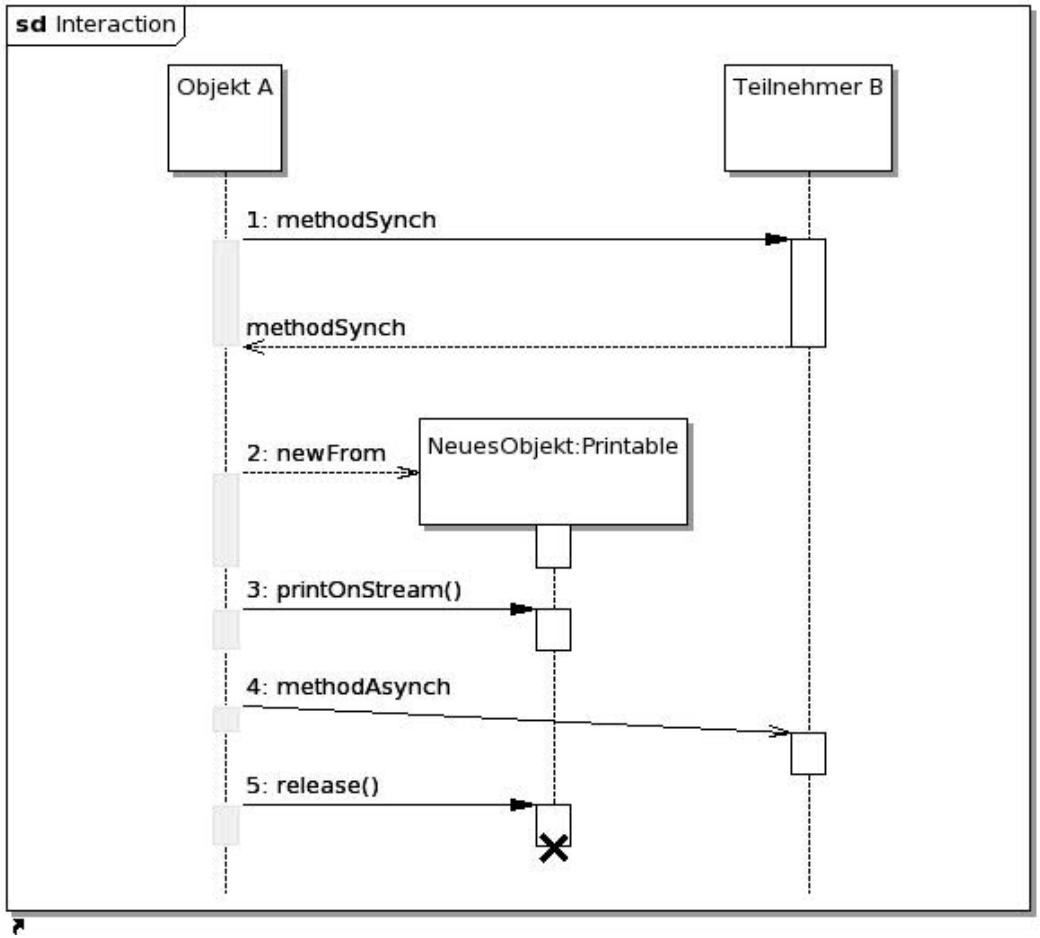
Sequenzendiagramm

- Stellt ein oder mehrere Szenarien als *eine Interaktion* dar
- Objekte, Lebenslinien, Nachrichten (Methoden)
- darüber hinaus: Kontrollfluß und Schachtelung
  - optionale Ausführung
  - alternative Ausführungen
  - Referenzen auf andere Interaktionsdiagramme
  - Sprungmarken

Sequenzendiagramm

- Stellt ein oder mehrere Szenarien als *eine Interaktion* dar
- Objekte, Lebenslinien, Nachrichten (Methoden)
- darüber hinaus: Kontrollfluß und Schachtelung
  - optionale Ausführung
  - alternative Ausführungen
  - Referenzen auf andere Interaktionsdiagramme
  - Sprungmarken

Sequenzdiagramme



Grundsätzliches

- Sehr gute Online-Hilfe zur Vorgehensweise
- Hyperlinks verketten Diagramme
- Together ist über FauxPas- Server erhältlich