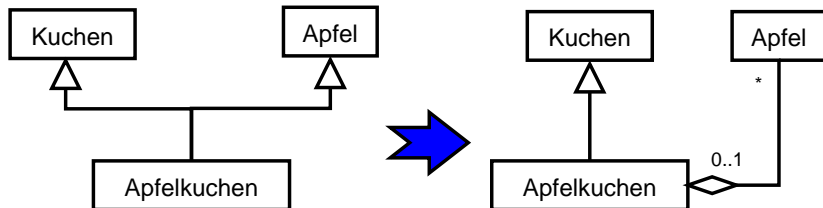


Liskovs Substitutionsprinzip (1988; 1994)

Definition

Liskovs Substitutionsprinzip: jede Instanz der Unterklasse kann immer und überall dort eingesetzt werden, wo Instanzen der Oberklasse auftreten.



Beschreibungsinhalt

- statische Systemaspekte
- Beschreibung der wesentlichen, unterscheidbaren Dinge eines Systems und ihrer Beziehungen

zentrale Modellierungskonstrukte:

- Klassen
- Assoziationen (Beziehungsklassen)
 - spezielle Assoziationen: Generalisierung und Aggregation/Komposition

Objektorientierte Modellierung

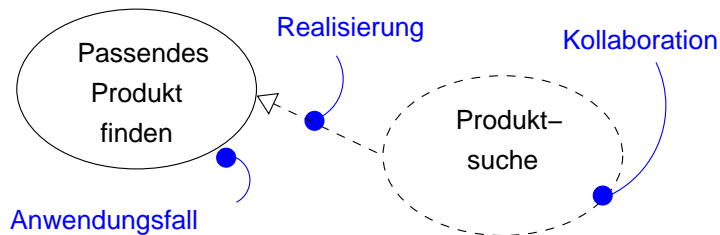
- Identifiziere Aktoren
- Beschreibe Anwendungsfälle
- Bestimme statisches Modell
 - Identifiziere Objekte
 - Identifiziere Eigenschaften der Objekte
 - Bestimme Assoziationen zwischen Objekten
 - Fasse Objekte zu Klassen zusammen
 - Ordne Klassen in Vererbungshierarchien ein
 - Bestimme Multiplizitäten der Assoziationen
- **Erstelle Verhaltensmodell**
 - **Identifiziere Ereignisse und modelliere Interaktionen in Anwendungsfällen**
 - Identifiziere Verhalten der Objekte
 - Beschreibe das Verhalten (Vor- und Nachbedingungen)

- textuelle Szenario-Beschreibungen
- Interaktionsdiagramme
- Aktivitätsdiagramme
- Zustandsdiagramme

Textuelle Beschreibung von Anwendungsfällen

- Akteure:
 - Kunde und Verkäufer
- Vorbedingung:
 - Kunde möchte ergänzenden Artikel kaufen
 - Kunde und Verkäufer sind im Laden
 - PDA und Ladenrechner sind in Betrieb
- Nachbedingung:
 - Kunde hat Artikel ausgewählt
- Ablauf:
 - *(siehe oben)*
- Varianten:
 - Kunde findet keinen passenden Artikel
→ Verkäufer berät Kunde
 - keine Verbindung zwischen PDA und Ladenrechner
→ Verkäufer berät Kunde

UML-Notation für Anwendungsfälle (OMG)



Beschreibungsinhalt:

- dynamische Systemaspekte
- exemplarische Beschreibung von Interaktionsabfolgen zwischen kollaborierenden Objekten (Inter-Objektverhalten)

zentrale Modellierungskonstrukte:

- Objekte: „Dinge“ mit eigener Identität
- Nachrichten
 - beschreiben den Austausch von Informationen zwischen Objekten zum Auslösen von Aktivitäten
 - sind Signale/Ereignisse oder Methodenaufrufe

Anwendung

- Präzisierung von Szenarien (exemplarische Folgen von Aktivitäten)
- Protokollierung des Nachrichtenaustausches
- Erhebung der von einzelnen Objekten bereit gestellten Funktionalität (Dienste, Methoden)

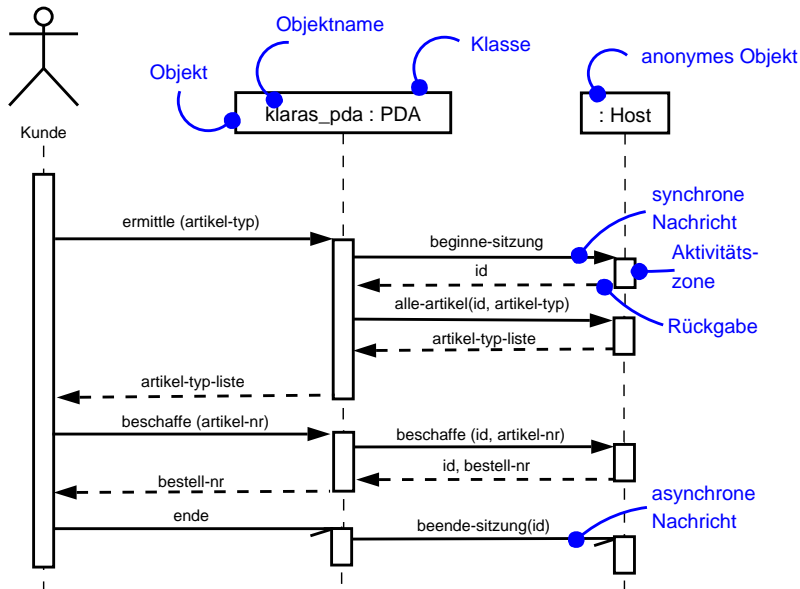
Varianten

- Sequenzdiagramme
 - betonen zeitlichen Ablauf
- Kollaborationsdiagramme
 - betonen Objektstruktur

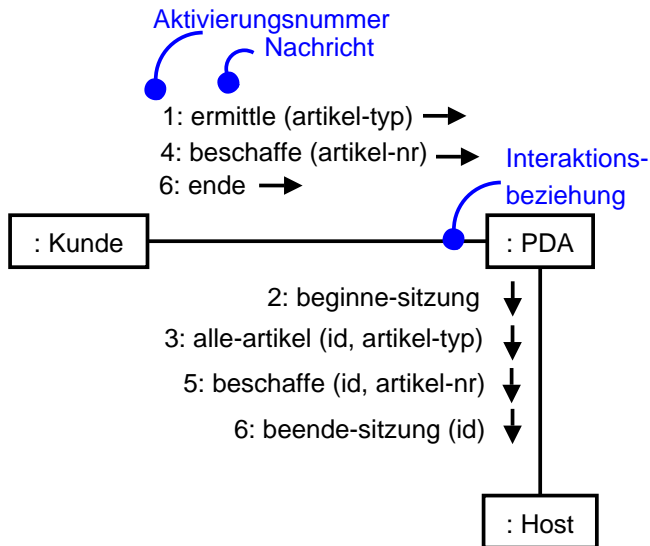
Grenzen:

- beschreiben Verhalten nur beispielhaft und nicht vollständig

UML-Sequenzdiagramme (OMG)



UML-Kollaborationsdiagramme (OMG)



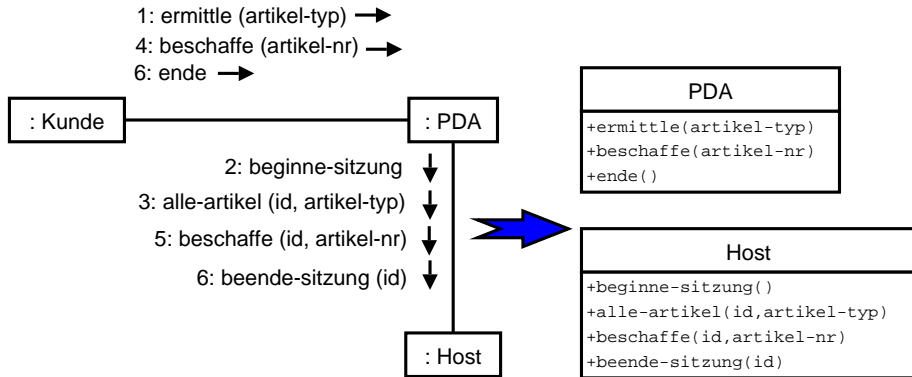
Objektorientierte Modellierung

- Identifiziere Aktoren
- Beschreibe Anwendungsfälle
- Bestimme statisches Modell
 - Identifiziere Objekte
 - Identifiziere Eigenschaften der Objekte
 - Bestimme Assoziationen zwischen Objekten
 - Fasse Objekte zu Klassen zusammen
 - Ordne Klassen in Vererbungshierarchien ein
 - Bestimme Multiplizitäten der Assoziationen
- Erstelle Verhaltensmodell
 - Identifiziere Ereignisse und modelliere Interaktionen in Anwendungsfällen
 - **Identifiziere Verhalten der Objekte**
 - Beschreibe das Verhalten (Vor- und Nachbedingungen)

Bemerkung:

- Interaktionsdiagramme beschreiben exemplarische Interaktionsszenarien eines Systems aus dynamischer Sicht
- Klassendiagramme beschreiben diese Systeme aus statischer, schematischer Sicht
- Interaktionsdiagramme dienen als Ausgangspunkt zur Ergänzung und Überprüfung von Klassendiagrammen

Interaktionen → Methoden



Interaktionen

- beschreiben das Wechselspiel zwischen Akteuren über Botschaften/Methoden¹
- liefern Methoden für die modellierten Klassen
- beschreiben jedoch **nicht** die Methoden selbst

¹Methoden beschreiben hier abstraktes Verhalten im Kontext der Anforderungen; sind noch nicht notwendigerweise die Methoden im Sinne der Implementierung (führen letztlich aber zu diesen hin).

Objektorientierte Modellierung

- Identifiziere Aktoren
- Beschreibe Anwendungsfälle
- Bestimme statisches Modell
 - Identifiziere Objekte
 - Identifiziere Eigenschaften der Objekte
 - Bestimme Assoziationen zwischen Objekten
 - Fasse Objekte zu Klassen zusammen
 - Ordne Klassen in Vererbungshierarchien ein
 - Bestimme Multiplizitäten der Assoziationen
- Erstelle Verhaltensmodell
 - Identifiziere Ereignisse und modelliere Interaktionen in Anwendungsfällen
 - Identifiziere Verhalten der Objekte
 - **Beschreibe das Verhalten (Vor- und Nachbedingungen)**

Beschreibung:

- **Parameter:** Eingabe und Ausgabe
- **Vorbedingung:** beschreibt die Annahmen der Methode, die gelten müssen, damit sie ausgeführt werden kann
- **Nachbedingung:** beschreibt das Resultat der Methode
- **Fehlerbedingungen:** Verletzung der Vorbedingungen und Fehler, die während der Ausführung auftreten können
- **Verhalten in Fehlersituationen:** Nachbedingung für jeden aufgetretenen Fehler
- **Reaktionszeit:** Maximale Dauer, bis Resultat vorliegt (sowohl im Normal- als auch im Fehlerfall)

Beispiel Sortierung und Ausgabe der Artikelliste:

- **Parameter:**

- Eingabe: Artikelliste, Attribut
- Ausgabe: Artikelliste'

- **Vorbedingung:**

- Attribut kommt bei allen Artikeln der Artikelliste vor

- **Nachbedingung:**

- Artikelliste' ist sortiert, d.h.:
 $\forall i : 1 \leq i < \text{len}(\text{Artikelliste}') \Rightarrow \text{element}(\text{Artikelliste}', i) \leq_{\text{Attribut}} \text{element}(\text{Artikelliste}', i + 1)$
- Artikelliste' ist eine Permutation von Artikelliste

- **Fehlerbedingungen:** keine, außer Vorbedingung nicht erfüllt

- **Verhalten in Fehlersituationen:**

- Artikelliste wird zurückgegeben
- Fehlermeldung wird ausgegeben

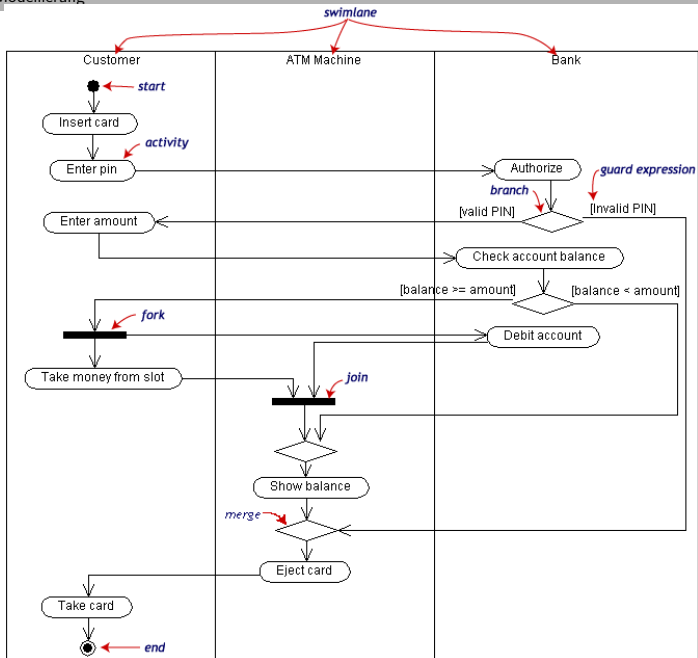
- **Reaktionszeit:** $n \cdot \log(n) \cdot 0.001$ [sec], wobei n die Länge der Liste ist

Beschreibung der Benutzeroberfläche

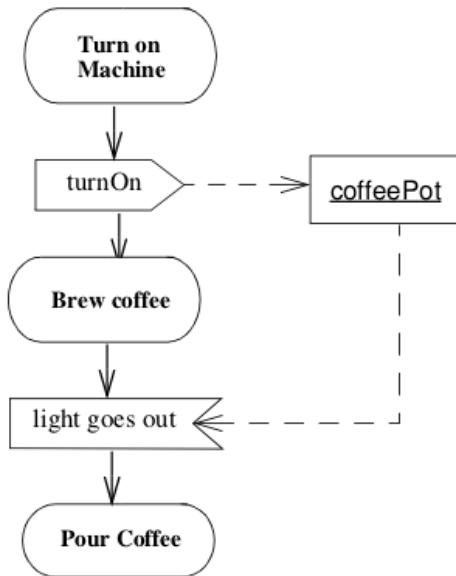


- 1 Grafik des Artikels in der Größe 40x40 Pixel
- 2 Artikeldaten (10pt-Schrift)
- 3 Attribut, nach dem sortiert wurde (rot, 12pt-Schrift)
- 4 Scrolling nach oben; es wird immer um einen Artikel nach oben geschoben
- 5 Scrolling nach unten; es wird immer um einen Artikel nach unten geschoben
- 6 Auswahl des Artikels; alternativ kann man mit dem Stift durch rasches doppeltes Anklicken einen Artikel auswählen

- Wurzeln: Flussdiagramme, Petrinetze
- modellieren klassenübergreifendes Verhalten (Kontrollfluss)
- beschreiben häufig Anwendungsfälle
- lt. UML-Standard „Zustandsautomat einer Berechnung“
- Einsatz empfohlen bei hauptsächlich intern ausgelösten Zustandsübergängen (einfacher Kontrollfluss)



Aktivitätsdiagramme: Signale



- modellieren Objektlebenszyklus: Verhalten eines Objekts in Bezug auf die verfügbaren Operationen seiner Klasse

- modellieren Objektlebenszyklus: Verhalten eines Objekts in Bezug auf die verfügbaren Operationen seiner Klasse
- gehen zurück auf Zustandsautomaten nach Harel (1987)

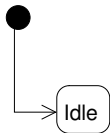
- modellieren Objektlebenszyklus: Verhalten eines Objekts in Bezug auf die verfügbaren Operationen seiner Klasse
- gehen zurück auf Zustandsautomaten nach Harel (1987)
- **Zustand**: Menge von Attributwerten eines Objekts

- modellieren Objektlebenszyklus: Verhalten eines Objekts in Bezug auf die verfügbaren Operationen seiner Klasse
- gehen zurück auf Zustandsautomaten nach Harel (1987)
- Zustand: Menge von Attributwerten eines Objekts
- **Transition**: Zustandsänderung

- modellieren Objektlebenszyklus: Verhalten eines Objekts in Bezug auf die verfügbaren Operationen seiner Klasse
- gehen zurück auf Zustandsautomaten nach Harel (1987)
- Zustand: Menge von Attributwerten eines Objekts
- Transition: Zustandsänderung
- geeignet auch zur Beschreibung von Anwendungsfällen und Protokollen

- modellieren Objektlebenszyklus: Verhalten eines Objekts in Bezug auf die verfügbaren Operationen seiner Klasse
- gehen zurück auf Zustandsautomaten nach Harel (1987)
- Zustand: Menge von Attributwerten eines Objekts
- Transition: Zustandsänderung
- geeignet auch zur Beschreibung von Anwendungsfällen und Protokollen
- erlauben Verschachtelung von **Zuständen** und **Automaten**

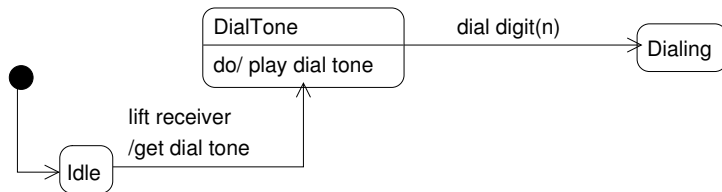
Zustandsautomatendiagramme: Beispiel



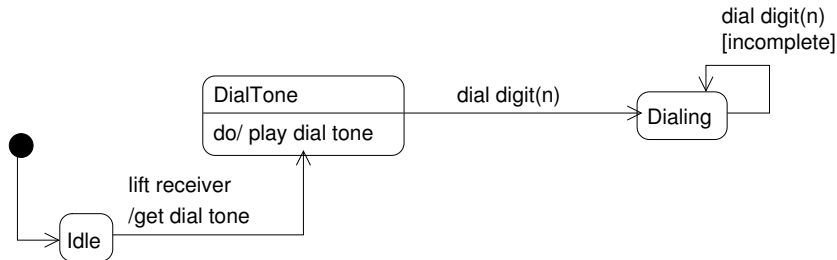
Zustandsautomatendiagramme: Beispiel



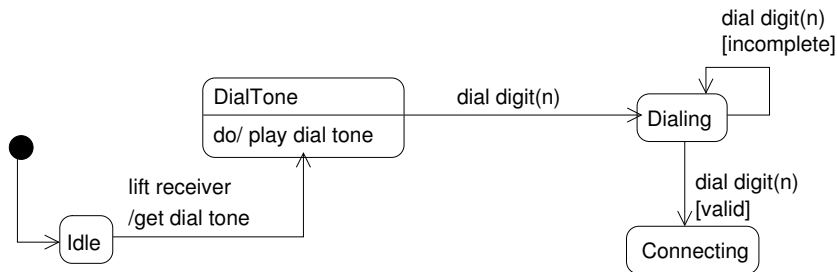
Zustandsautomatendiagramme: Beispiel



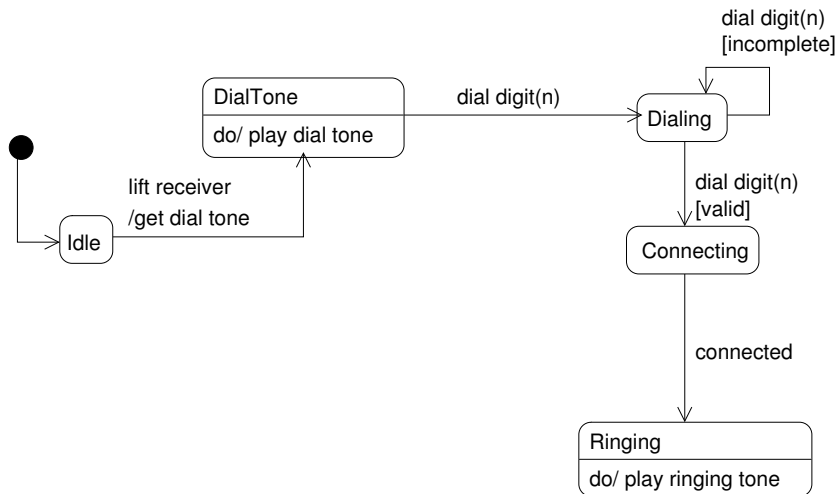
Zustandsautomatendiagramme: Beispiel



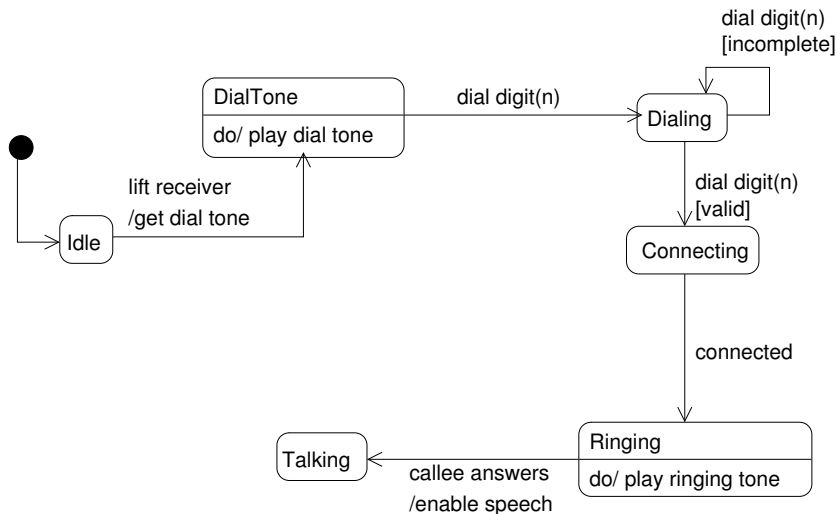
Zustandsautomatendiagramme: Beispiel



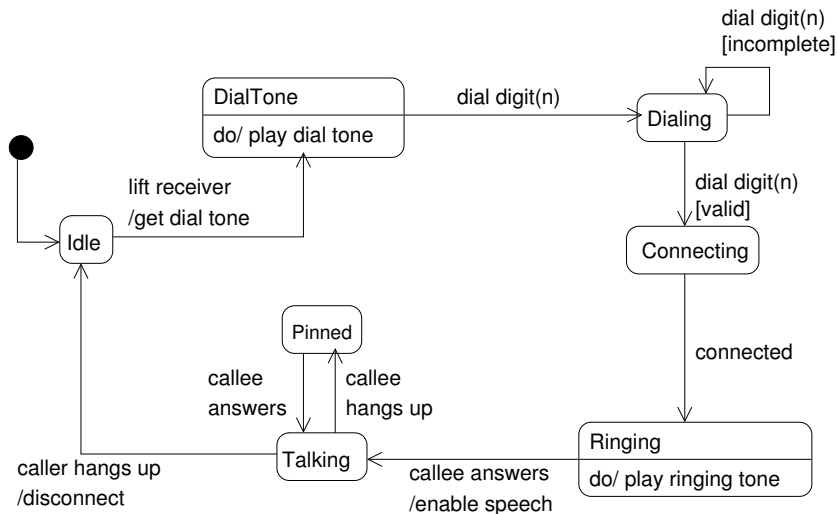
Zustandsautomatendiagramme: Beispiel



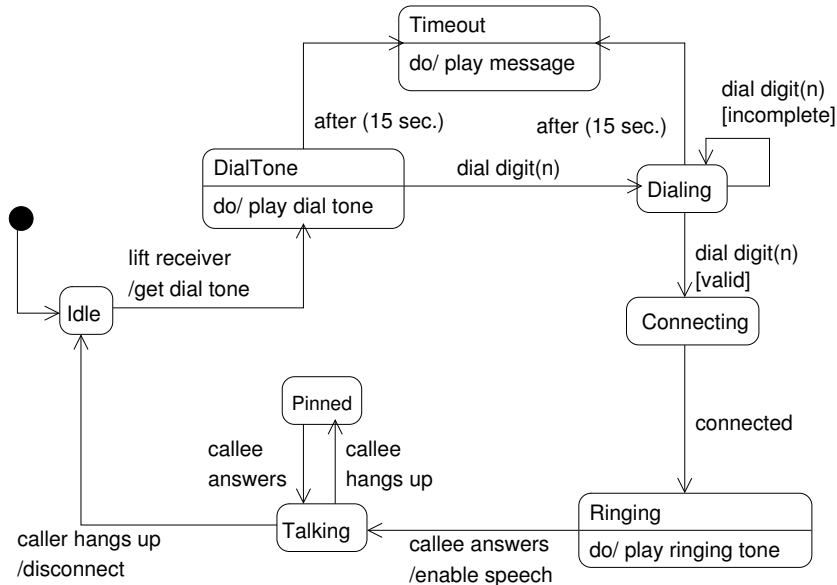
Zustandsautomatendiagramme: Beispiel



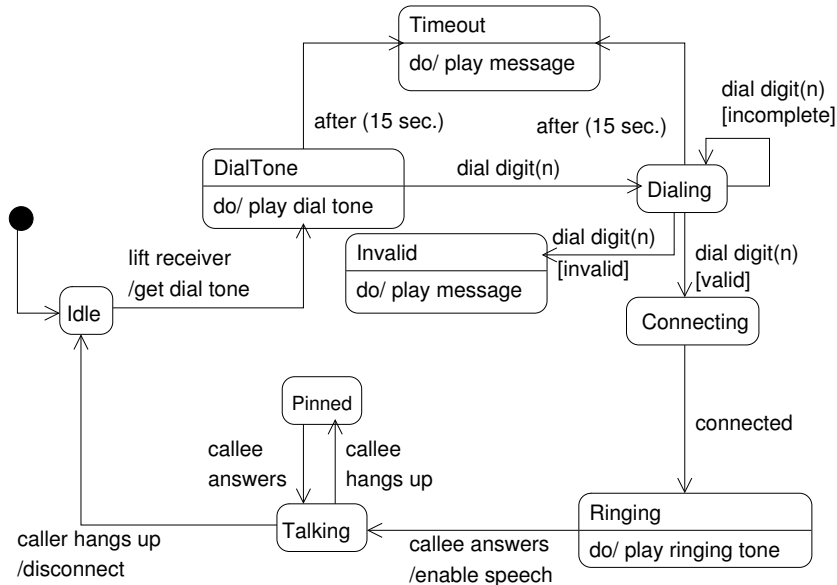
Zustandsautomatendiagramme: Beispiel



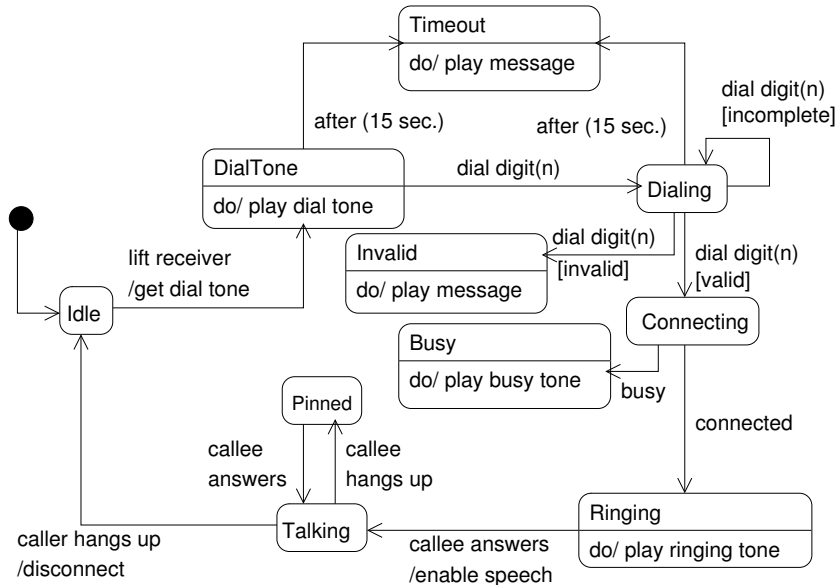
Zustandsautomatendiagramme: Beispiel

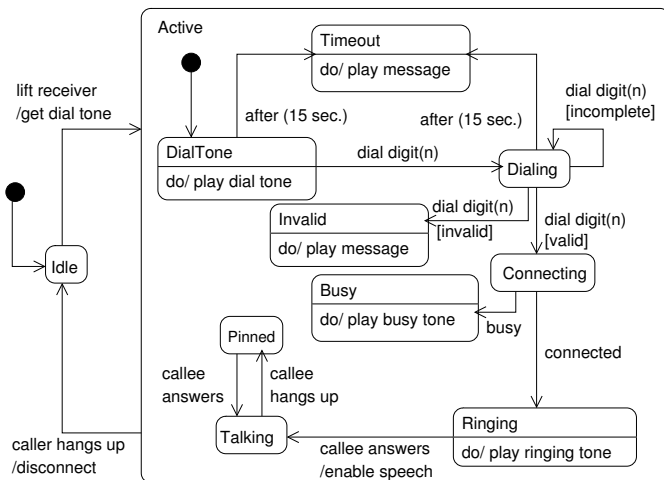


Zustandsautomatendiagramme: Beispiel



Zustandsautomatendiagramme: Beispiel





Zustände

- einfach/zusammengesetzt