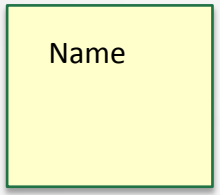
UML - Anwendungsfalldiagramme

Motivation: Warum Anwendungsdiagramme?

- ✓ Anwendungsfälle konzentrieren sich auf das fundamentale Problem bei der Entwicklung eines Systems, der Entwicklung einer Lösung für den Kunden bzw. Anwender, die der Kunde bzw. der Anwender auch gewünscht hat.
- ✓ Anwendungsfälle repräsentieren die Anforderungen der Kunden

Regeln: System

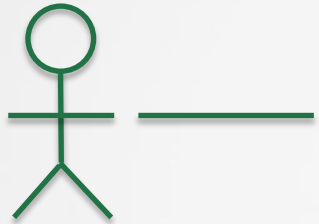


- ✓ Das Rechteck stellt das geplante System dar.
- ✓ Der Name gibt den Namen des Systems an.
- ✓ Ein Anwendungsfall Diagramm kann auch mehrere Systeme enthalten.
- ✓ Dadurch kann ein System in Teilsysteme gegliedert werden.

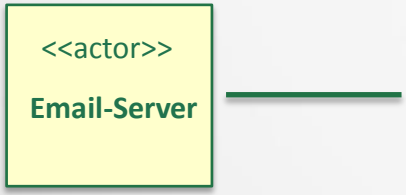
Regeln: Anwendungsfall

- ✓ Eine Ellipse stellt einen Anwendungsfall des Systems dar.
- ✓ Ein Anwendungsfall ist ein in sich abgeschlossener Vorgang, der für einen oder mehrere Akteure ein beobachtbares Ergebnis liefert.
- ✓ Er beschreibt aus Sicht der Akteure, welche Leistungen das System für den Anwender zur Verfügung stellt.
- ✓ Ein Anwendungsfall stellt somit einen Teil der Gesamtfunktionalität des Systems dar.
- ✓ In UML 2.0 kann auch ein Rechteck, das mit einer Ellipse markiert wird, als Anwendungsfall-Symbol verwendet werden.
- ✓ Der Name kann innerhalb oder außerhalb des Symbols stehen.

Regeln: Akteur

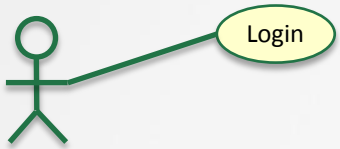


Benutzer



- ✓ Ein Akteur ist ein Element, das nicht zum geplanten System gehört.
- ✓ Er kann eine Person sein, die auf das System zugreift, oder ein anderes System, das mit dem geplanten System kommuniziert
  Die UML erlaubt mehrere Symbole zur Darstellung eines Akteurs.
  - Er kann als Strichmännchen dargestellt werden(menschlich, aktiv).
  - Es ist optional erlaubt ein Klassensymbol zu verwenden, das mit dem Stereotyp <<actor>> markiert wird(nicht-menschlich, passiv)..
  - Zusätzlich können eigene Symbole verwendet werden, um nicht menschliche Akteure darzustellen.
- Ein konkrete Person kann mehrere Rollen z. B. Benutzer und Servicetechniker eines Systems sein

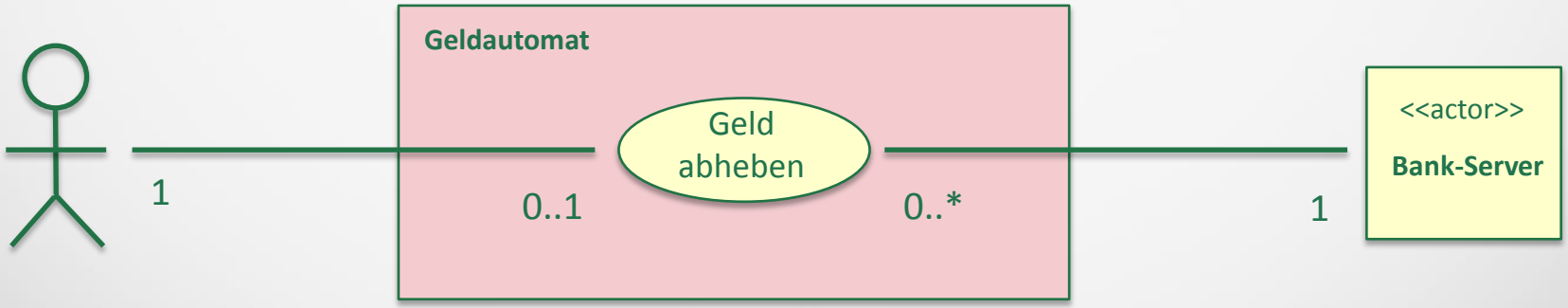
Regeln: Assoziation I



- ✓ Eine Linie stellt eine Assoziation zwischen einem Akteur und einem Anwendungsfall dar.
- ✓ Sie beschreibt den Zugriff des Akteurs auf die Funktionalität, die das System in diesem Anwendungsfall zur Verfügung stellt, bzw. eine Antwort des Systems an einen Akteur.

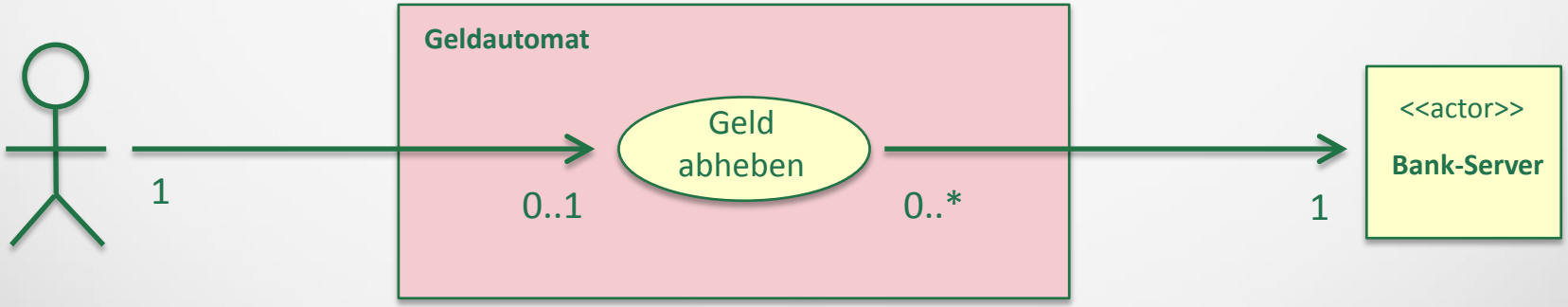
Regeln: Assoziation II

- ✓ Die Multiplizität auf Seite des Anwendungsfalles gibt an, wie oft dieser Anwendungsfall vom Akteur gleichzeitig ausgeführt werden darf.
- ✓ Wenn keine Angabe gemacht wird, ist die Multiplizität immer 0..1.
- ✓ Auf der Seite des Akteurs bedeutet die Multiplizität, wie viele Akteure der angegebenen Rolle am Anwendungsfall beteiligt sein müssen bzw. können.
- ✓ Wenn keine Angabe gemacht wird, ist die Multiplizität 1..1 oder 1



Regeln: Navigationsangaben

- ✓ Optional
- ✓ Keine Datenflussrichtung: Geben den Initiator der Kommunikation an.
- ✓ Beschreibung: Aktiver und passiver Teil
  - ✓ Akteur navigiert zu einem Anwendungsfall: Akteur ist der Aktive und stößt den Anwendungsfall an
  - ✓ Anwendungsfall navigiert zum Akteur: Akteur ist der Passive und wird vom Anwendungsfall benötigt/aufgefordert teilzunehmen

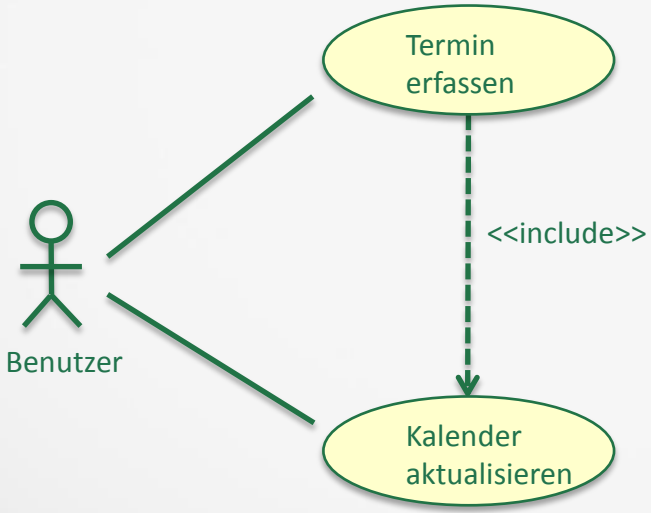


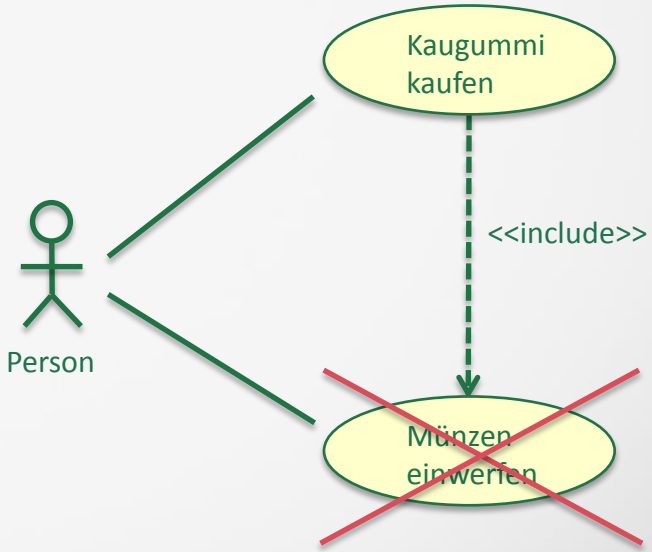
Regel: Include-Assoziation I

- ✓ Bei der include Beziehung verwendet ein Anwendungsfall die Funktionalität, die ei anderer Anwendungsfall zur Verfügung stellt.
- ✓ Der includierte Anwendungsfall wird immer ausgeführt.
- ✓ Der Anwendungsfall, von dem die Verbindungslinie ausgeht, schließt den Anwendungsfall ein auf den die Verbindungslinie zeigt.
- ✓ Irgendwann mitten in der Ausführung des linken Anwendungsfall beginnt der Anwendungsfall zu laufen. Wenn der rechte Anwendungsfall B dann beendet ist, setzt die Ausführung des linken Anwendungsfall fort.

Regeln: Include-Assoziation II

- ✓ Voraussetzung für die Erstellung des Anwendungsfalles bei der Modellierung: Der inkludierte Anwendungsfall kann selbstständig ausgeführt werden.



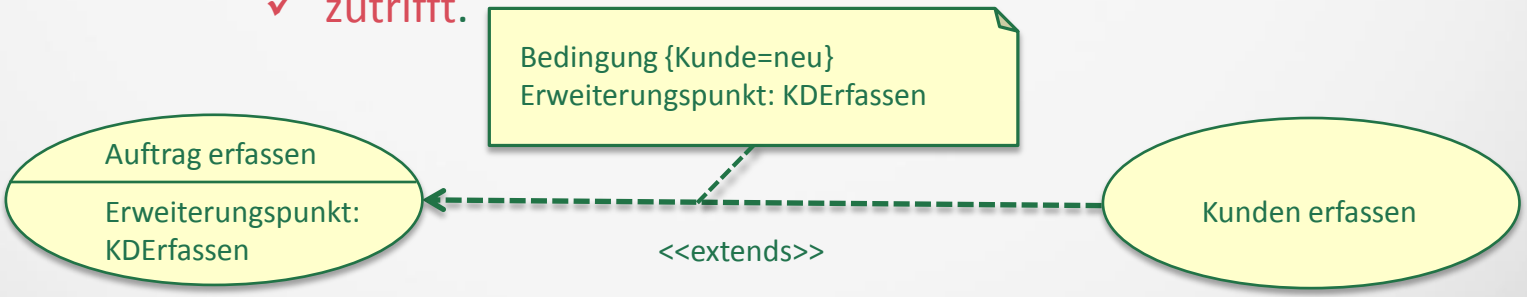


Regeln: Extend-Assoziation

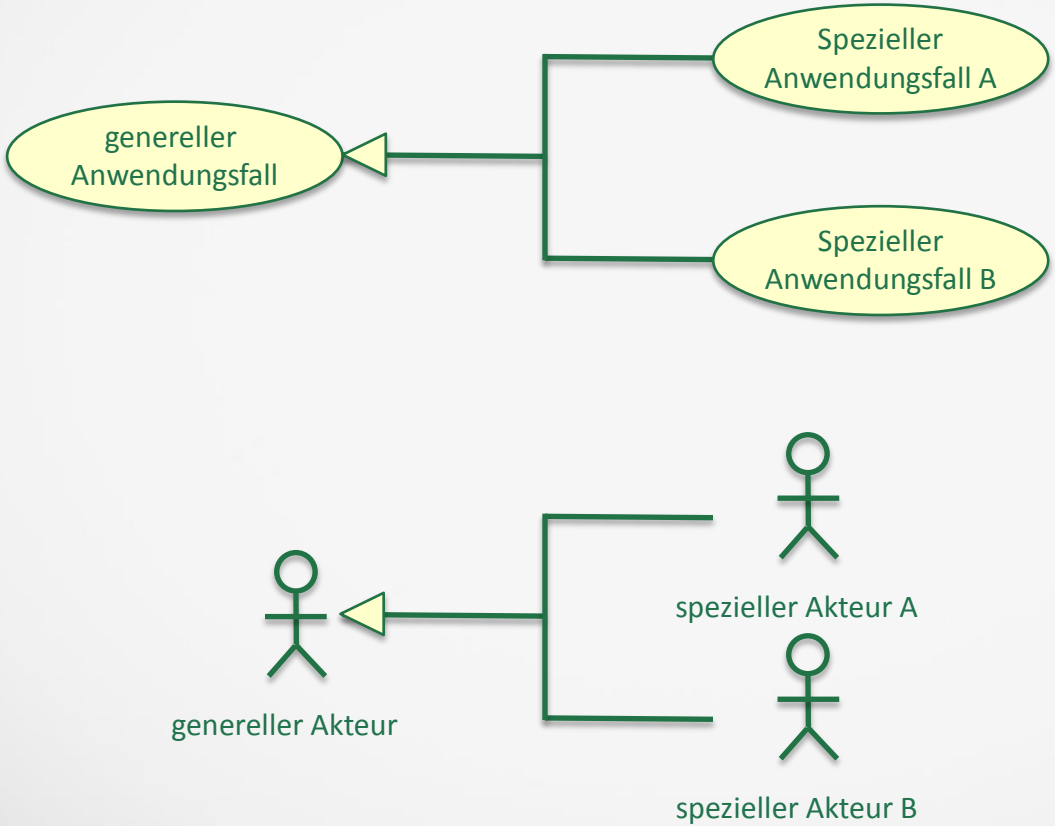
- ✓ Die extends - Beziehung beschreibt die Erweiterung der Funktionalität eines Anwendungsfalls durch einen anderen Anwendungsfall.
- ✓ Man kann dadurch optionales Verhalten beschreiben, bzw. Funktionen modellieren, die nur unter bestimmten Bedingungen ausgeführt werden.
- ✓ Der Anwendungsfall, von dem die Verbindungslinie ausgeht, erweitert möglicherweise den Anwendungsfall, auf den die Verbindungslinie zeigt.

Regeln: Extension point

- ✓ Ein extension point gibt bei einer extend-Beziehung den Punkt an, an dem der erweiternde Anwendungsfall im erweiterten Anwendun eingehängt ("aufgerufen") wird.
- ✓ Es kann eine Bedingung für den Aufruf angegeben werden.
- ✓ In diesem Beispiel erweitert der Anwendungsfall B den Anwendungsfall A.
- ✓ Er wird an dem Erweiterungspunkt "ruftB" in A eingehängt.
- ✓ Der Aufruf erfolgt nur, wenn die Bedingung "wenn Bedingung"
- ✓ zutrifft.

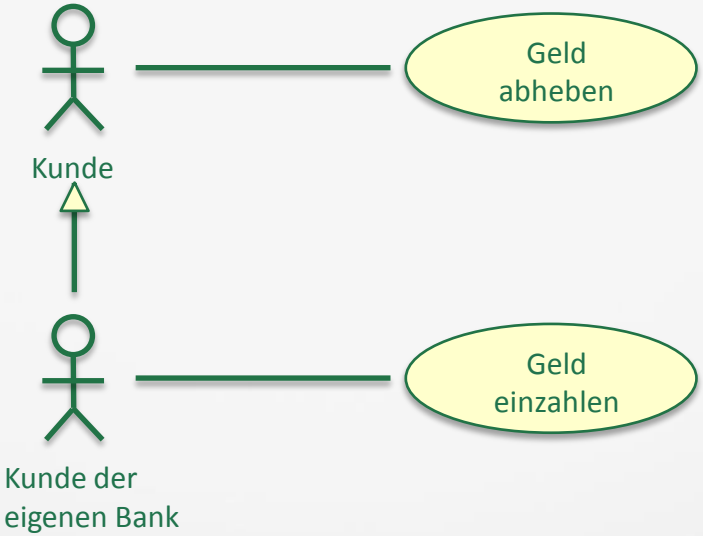


Regeln: Generalisierung – Spezialisierung I



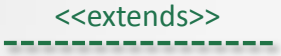
Regeln: Generalisierung – Spezialisierung II

- ✓ Generalisierungsbeziehungen werden auch eingesetzt, um Funktionalitäten allgemein und abstrakt zu beschreiben.
- ✓ Mancher Anwendungsfall ist abstrakt und kann selbst nicht ausgeführt werden!
- ✓ Bei dieser Art der Verbindung spricht man auch von einer „is-a“ Beziehung, da alles vom generelleren Element „geerbt“ wird.



Regeln: Verbindungslinien(Assoziationen)

verwendet

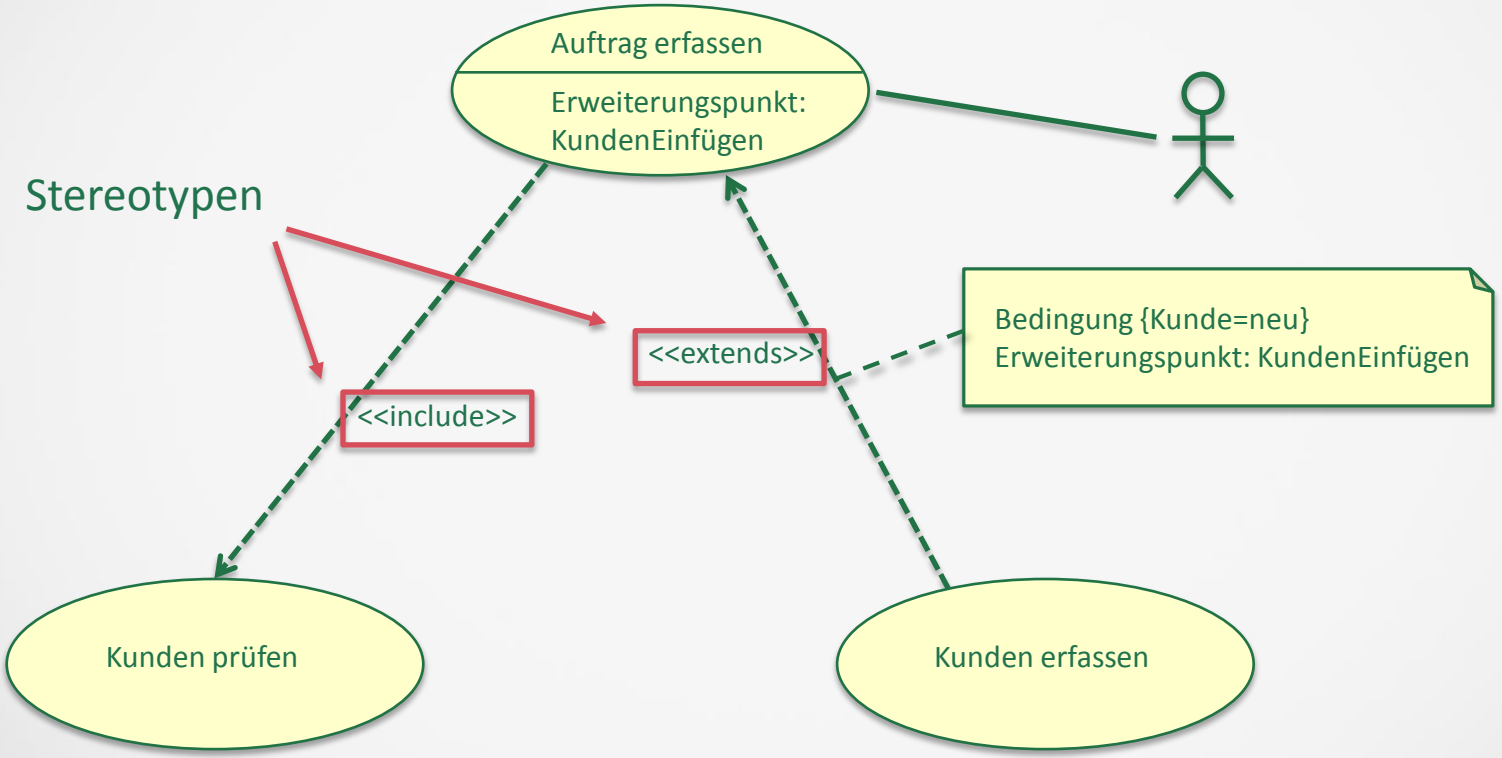


erweitert

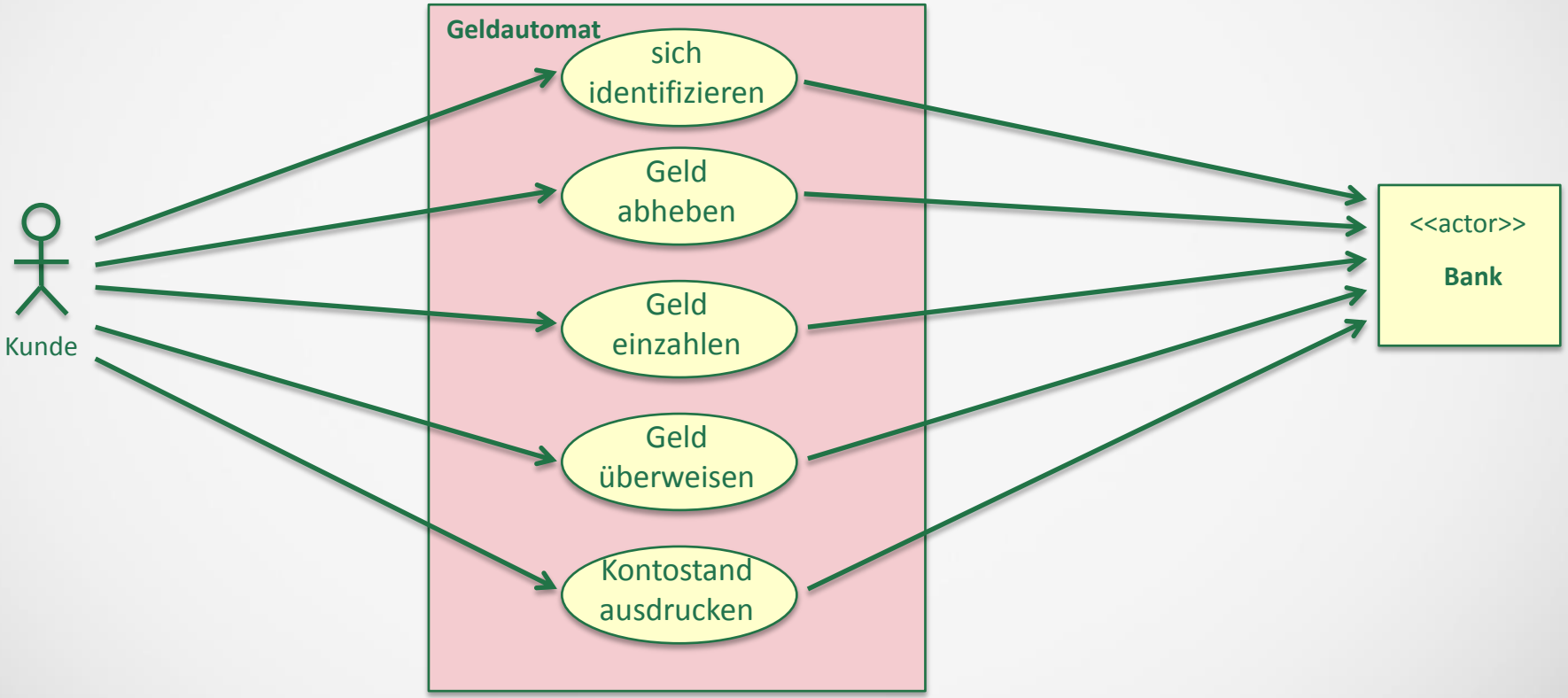
enthält

generalisiert/spezialisiert

Beispiel I:


Auftrag erfassen
Erweiterungspunkt:
KundenEinfügen
Stereotypen
<<extends>>
<<include>>
Bedingung {Kunde=neu}
Erweiterungspunkt: KundenEinfügen
Kunden prüfen
Kunden erfassen

Beispiel II:


Geldautomat
sich identifizieren
Geld abheben
Geld einzahlen
Geld überweisen
Kontostand ausdrucken
Kunde
<<actor>>
Bank