

7



Anwendungsfalldiagramme

Anwendungsfälle (use cases) stellen in der UML die Funktionalität und Anforderungen eines Systems dar. Sie bestehen aus benannten Funktionalitäten (*Anwendungsfälle*), den Personen oder Sachen, die diese Funktionalität aufrufen (*Akteure*), und möglicherweise den Elementen, die für die Implementierung der Anwendungsfälle zuständig sind (*Subjekte*).

Anwendungsfälle

Anwendungsfälle stellen unterschiedliche Einzelfunktionalitäten eines Systems, einer Komponente oder einer Klasse dar. Jeder Anwendungsfall muss einen Namen haben; üblicherweise eine kurze Beschreibung der verlangten Funktionalität wie etwa Fehlerprotokoll anzeigen. Die UML bietet zwei Möglichkeiten, einen Anwendungsfall darzustellen: Die erste ist ein Oval mit dem Namen des Anwendungsfalls in der Mitte, wie in Abbildung 7-1 gezeigt.



Abbildung 7-1: Ein einfacher Anwendungsfall

Auch das Ovalsymbol eines Anwendungsfalles lässt sich wieder in Abschnitte unterteilen, um mehr Einzelheiten wie beispielsweise Erweiterungspunkte (siehe »Erweiterung von Anwendungsfällen«), eingebundene Anwendungsfälle (siehe »Enthaltene Anwendungsfälle«) oder bestimmte Einschränkungen anzuzeigen. Abbildung 7-2 zeigt ein Anwendungsfallsymbol mit einem Abschnitt, der Erweiterungspunkte enthält.

Doch diese Darstellung von Anwendungsfällen ermöglicht keine allzu detaillierten Abschnitte. Daher empfiehlt die UML die Classifier-Notation, wenn Sie mehr Einzelheiten des Anwendungsfalles anzeigen möchten. In diesem Fall stellen Sie den Anwendungsfall als Rechteck mit dem Ovalsymbol in der oberen rechten Ecke dar. Ganz oben schreiben Sie den Namen

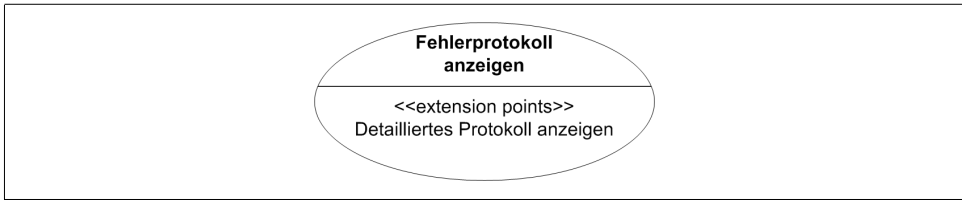


Abbildung 7-2: Anwendungsfall mit einem Abschnitt für Erweiterungspunkte

des Anwendungsfalls in Fettdruck hin. Darunter wird der Classifier nach Bedarf in Abschnitte unterteilt. Typische Namen für solche Abschnitte wären etwa *extension points* (Erweiterungspunkte) und *included use cases* (enthaltene Anwendungsfälle). Abbildung 7-3 zeigt denselben Anwendungsfall wie Abbildung 7-2, aber dieses Mal in der Classifier-Notation.

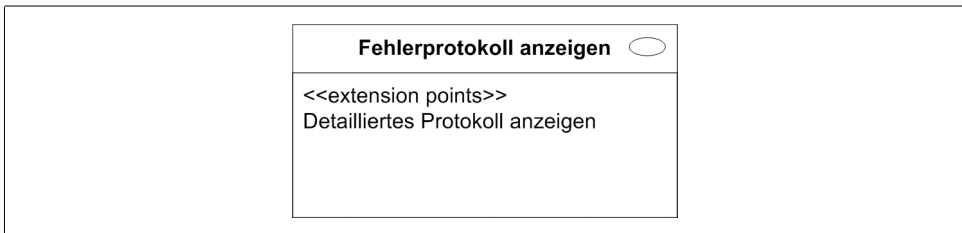


Abbildung 7-3: Anwendungsfall in der Classifier-Notation

In der UML gilt der Begriff *Anwendung* ausdrücklich nur für dieses UML-Element und seinen Namen. Die vollständige Dokumentation eines Anwendungsfalls ist eine *Instanziierung* dieses Anwendungsfalls. Das ist ein feiner Unterschied, doch er ermöglicht es, einen Anwendungsfall immer gerade so zu dokumentieren, dass seine Funktionalität am besten wiedergegeben wird, sei es in einem Textdokument, Zustandsautomaten, Interaktionsdiagramm, Aktivitätsdiagramm oder in irgendeiner anderen Form, die sich besonders gut eignet, dem Leser die Funktionalität in allen Einzelheiten verständlich zu machen.

Akteure

Ein Anwendungsfall muss von einer außenstehenden Person oder Sache instanziiert werden. Dieser Interessent wird als *Akteur* bezeichnet. Ein Akteur muss kein menschlicher Benutzer sein: Auch ein anderes externes System oder Element kann den Anwendungsfall auslösen (oder Empfänger seiner Ergebnisse sein) und wird somit als Akteur modelliert. Es ist zum Beispiel recht üblich, die Systemuhr als Akteur zu modellieren, der einen Anwendungsfall zu einem gegebenen Zeitpunkt oder Zeitintervall auslöst.

Auch für Akteure kennt die UML mehrere verschiedene Formen der Darstellung. Die erste ist ein Strichmännchen mit dem Namen des Akteurs darunter, wie in Abbildung 7-4 dargestellt.

Alternativ lässt sich auch ein Akteur als Classifier darstellen. In diesem Fall wird er wieder als Rechteck gezeichnet, mit dem Schlüsselwort *actor* am oberen Rand und seinem Namen in Fettdruck darunter. Da Akteure in der Regel Abschnitte haben, wird diese Darstellung nicht sehr häufig benutzt. Abbildung 7-5 zeigt einen Akteur in der Classifier-Notation.

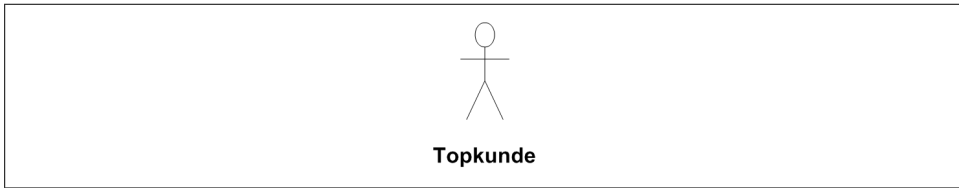


Abbildung 7-4: Ein Akteur als Strichmännchensymbol

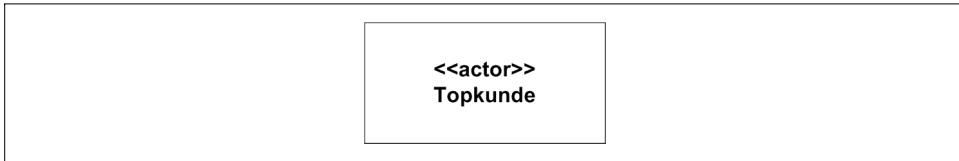


Abbildung 7-5: Ein Akteur in der Classifier-Notation

Wenn es hilfreich ist, können Sie auch eigene Symbole verwenden, um verschiedene Arten von Akteuren zu unterscheiden. Eine externe Datenbank könnte man zum Beispiel mit einem Datenbanksymbol darstellen, aber den Systemadministrator als Strichmännchen. Abbildung 7-6 zeigt genau diese Akteure.

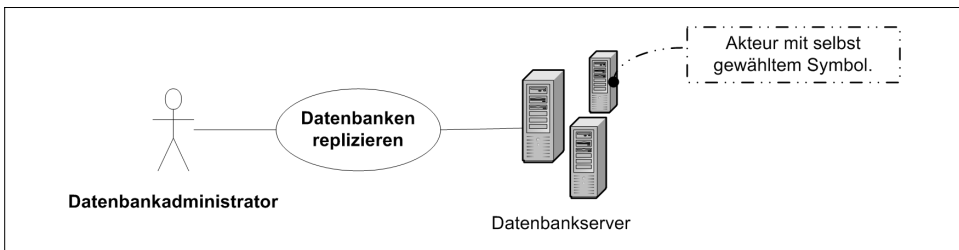


Abbildung 7-6: Akteur mit selbst gewähltem Symbol

Akteur/Anwendungsfall-Assoziationen

In der Regel verbinden Sie einen Akteur mit einem oder mehreren Anwendungsfällen. Eine Beziehung zwischen einem Akteur und einem Anwendungsfall bedeutet, dass entweder der Akteur den Anwendungsfall initiiert oder der Anwendungsfall dem Akteur Ergebnisse liefert oder beides. Eine Assoziation zwischen einem Akteur und einem Anwendungsfall ist eine durchgezogene Linie. Normalerweise werden Anwendungsfalldiagramme von links nach rechts gelesen, links mit den Akteuren, die die Anwendungsfälle initiieren, und rechts mit den Akteuren, die die Anwendungsfall-Ergebnisse bekommen. Doch je nach Modell oder Komplexität kann es sinnvoll sein, Akteure auf andere Weise zusammenzufassen. Abbildung 7-7 zeigt einen Akteur, der mit einem Anwendungsfall kommuniziert.

Auch wenn dies in der offiziellen UML-Spezifikation nicht vorgesehen ist, ist es üblich, an Assoziationslinien Richtungspeile anzubringen, um zu zeigen, wer die Kommunikation mit wem beginnt. Beachten Sie, dass die Pfeile nicht unbedingt eine Einschränkung für den Informationsfluss bedeuten; sie zeigen einfach von dem Initiator zu dem Empfänger der Kommunikation. Was geschieht, nachdem der Anwendungsfall ausgeführt wird, wird an anderer Stelle gesagt (siehe »Anwendungsfälle«). Abbildung 7-8 zeigt zwei Akteure und einen Anwendungsfall mit gerichteten Assoziationen.

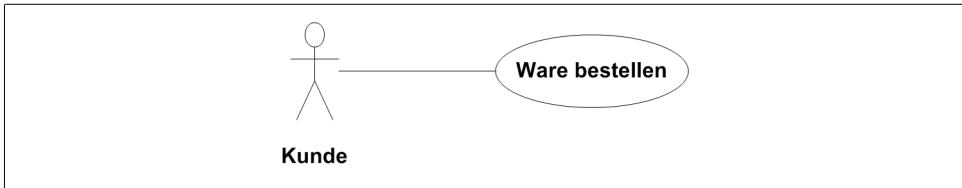


Abbildung 7-7: Ein Akteur ist mit dem Anwendungsfall »Ware bestellen« assoziiert.

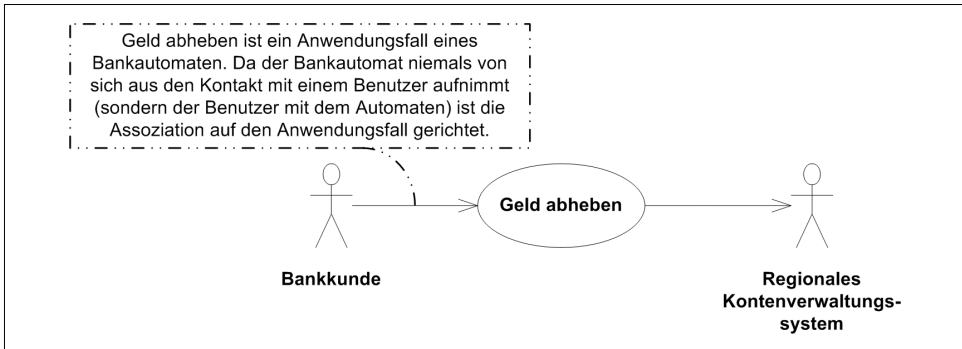


Abbildung 7-8: Beispiel für gerichtete Assoziationen zwischen Akteuren und einem Anwendungsfall

Systemgrenzen

Per Definition zeigen Anwendungsfälle die Funktionalität eines bestimmten Subjekts. All das, was von dem Subjekt nicht realisiert wird, liegt außerhalb der Systemgrenzen und sollte als Akteur modelliert werden. Diese Technik ist sehr nützlich, wenn es gilt, beim Entwurf eines Systems, eines Teilsystems oder einer Komponente den Gültigkeitsbereich und die Verantwortlichkeiten festzulegen. Wenn Sie beispielsweise ein System für einen Bankautomaten modellieren, drehen sich die Entwurfsdiskussionen letztlich um die Details des Backend-Banksystems. Ein Anwendungsfallmodell mit genau definierten Systemgrenzen würde das Banksystem als Akteur sehen und daher außerhalb des Problembereichs ansiedeln.

Systemgrenzen in einem generischen Sinn werden mit einem einfachen Rechteck mit dem Namen des Systems am oberen Rand dargestellt. Abbildung 7-9 zeigt die Systemgrenzen des im vorigen Absatz erwähnten Bankautomaten.

Funktionalität mithilfe von Akteuren definieren

Akteure müssen nicht unbedingt eins-zu-eins auf physische Entitäten abbildbar sein; eigentlich brauchen sie noch nicht einmal physische Entitäten zu sein. In der UML können Akteure auch Rollen potenzieller Systemnutzer darstellen. Unter Umständen ist der Systemadministrator der einzige *physische* Benutzer eines Systems, aber dieser Administrator kann viele Funktionen haben. Es mag hilfreich sein, das System aus der Perspektive eines Datenbankadministrators, Backupadministrators, Deployment-Administrators usw. zu betrachten. Bei genauer Betrachtung der verschiedenen Rollen der Akteure, die ein System später benutzen werden, entdeckt man oft Anwendungsfälle, die sonst gar nicht erkannt worden wären. Abbildung 7-10 zeigt ein Diagramm mit drei Typen von Administratoren und Beispiel-Anwendungsfällen.

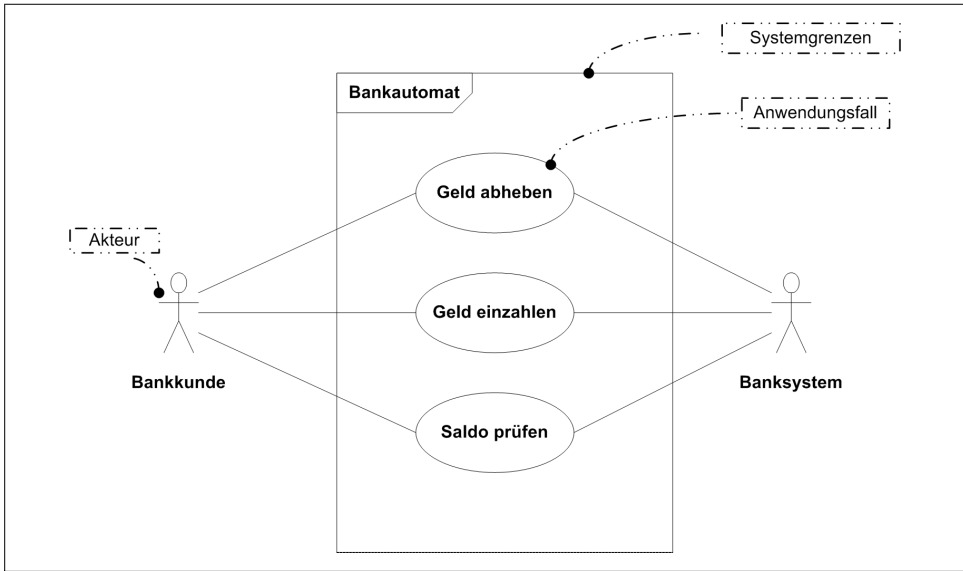


Abbildung 7-9: Das Anwendungsfalldiagramm zeigt die Systemgrenzen eines Banksystems.

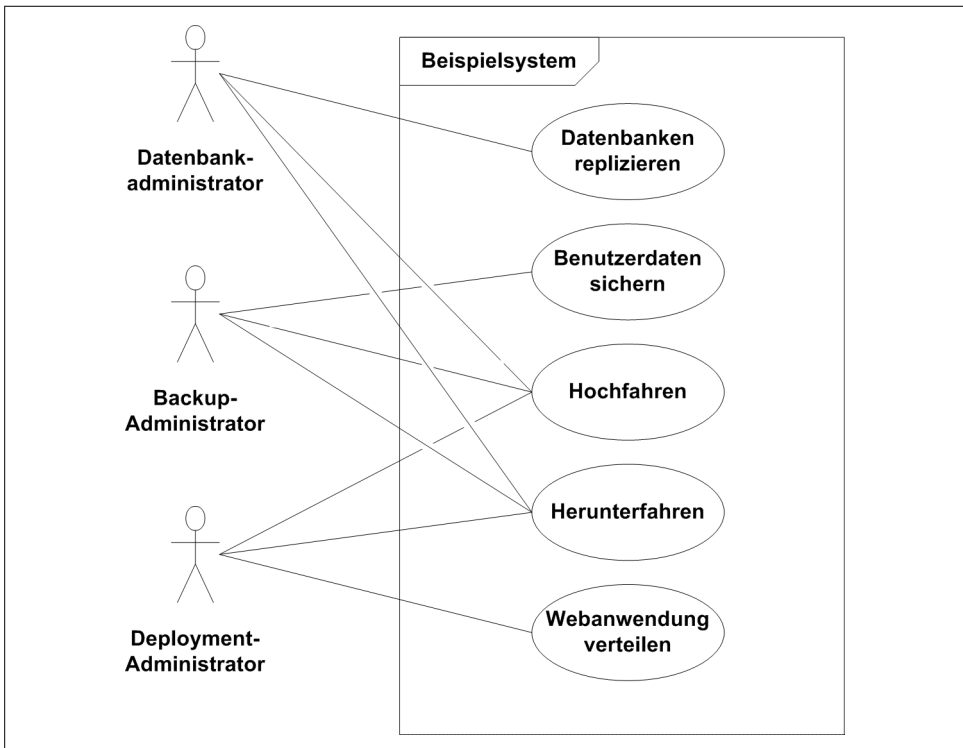


Abbildung 7-10: Spezialisiertere Versionen eines Akteurs helfen, die benötigte Funktionalität zu finden.

Fortgeschrittene Anwendungsfallmodellierung

Wie für andere Classifier bietet die UML auch für Anwendungsfälle und Akteure Mechanismen zur Erweiterung und Wiederverwendung. Durch *Generalisierung* können Sie die Fähigkeiten eines Akteurs ausweiten oder ganze Anwendungsfälle ersetzen. Sie können Elemente, die mehreren Anwendungsfällen gemeinsam sind, als *enthaltene (included)* Anwendungsfälle ausgliedern, oder Sie können Basis-Anwendungsfällen durch *Erweiterung (extension)* etwas hinzufügen.

Generalisierung von Akteuren und Anwendungsfällen

Auch wenn es nicht explizit in der Spezifikation steht: Akteure und Anwendungsfälle können, wie viele andere Classifier, auch generalisiert werden. Die Generalisierung von Akteuren dient normalerweise dazu, gemeinsame Anforderungen aus verschiedenen Akteuren herauszuholen, um die Modellierung zu vereinfachen. So zeigt zum Beispiel Abbildung 7-10 mehrere Administratoren und die Anwendungsfälle, die sie aufrufen müssen. Es kann einen Datenbankadministrator, einen Backup-Administrator und einen Deployment-Administrator geben, deren Bedürfnisse zwar jeweils ein wenig unterschiedlich sind, aber sich im Großen und Ganzen überschneiden. Nun extrahieren Sie aus diesen einen Systemadministrator-Akteur, in dem Sie die gemeinsame Funktionalität festhalten, und erstellen dann Spezialisierungen von ihm, um die jeweils einzigartigen Bedürfnisse der einzelnen Akteure zu modellieren.

Eine Akteur-Generalisierung stellen Sie wie jeden anderen Classifier dar: mit einer durchgezogenen Linie, deren geschlossene Pfeilspitze von dem spezialisierten Akteur zu dem Basisakteur zeigt. Abbildung 7-11 zeigt dieselben Informationen wie Abbildung 7-10, aber in einem wesentlich leserfreundlicheren Diagramm.

Auch Anwendungsfälle können generalisiert werden. In der Regel wird dies getan, um eine auf höherer Ebene angesiedelte Funktionalität darzustellen, die das System benötigt, ohne in die Einzelheiten zu gehen. Spezialisierungen eines solchen allgemeinen Anwendungsfalls führen die konkrete Funktionalität ein. Ein generischer Anwendungsfall könnte zum Beispiel Identität des Passagiers überprüfen sein, während Spezialisierungen dieses Anwendungsfalls Fingerabdruck des Passagiers überprüfen und RFID-Tag des Passagiers überprüfen lauten könnten. Es ist wichtig festzuhalten, dass auch bei einer Anwendungsfallgeneralisierung immer noch über Funktionalität geredet wird und nicht etwa über die Implementierung. Spezialisierungen eines Anwendungsfalls dürfen nie dazu da sein, *dieselbe Funktionalität* auf verschiedene Weisen zu implementieren, sondern nur, um *unterschiedliche Funktionalität* darzustellen.

Generalisierung wird bei Anwendungsfällen genau wie bei Akteuren dargestellt: mit einer durchgezogenen Linie, deren geschlossene Pfeilspitze vom speziellen auf den allgemeinen Anwendungsfall zeigt. Wenn der allgemeine Anwendungsfall eine abstrakte Funktionalität darstellt (also ein funktionales Konzept, ohne dass gesagt wird, was der Benutzer auf welche Weise tut), schreiben Sie den Namen des Anwendungsfalls kursiv. Abbildung 7-12 zeigt die Überprüfen-Anwendungsfälle und ihre Beziehungen.

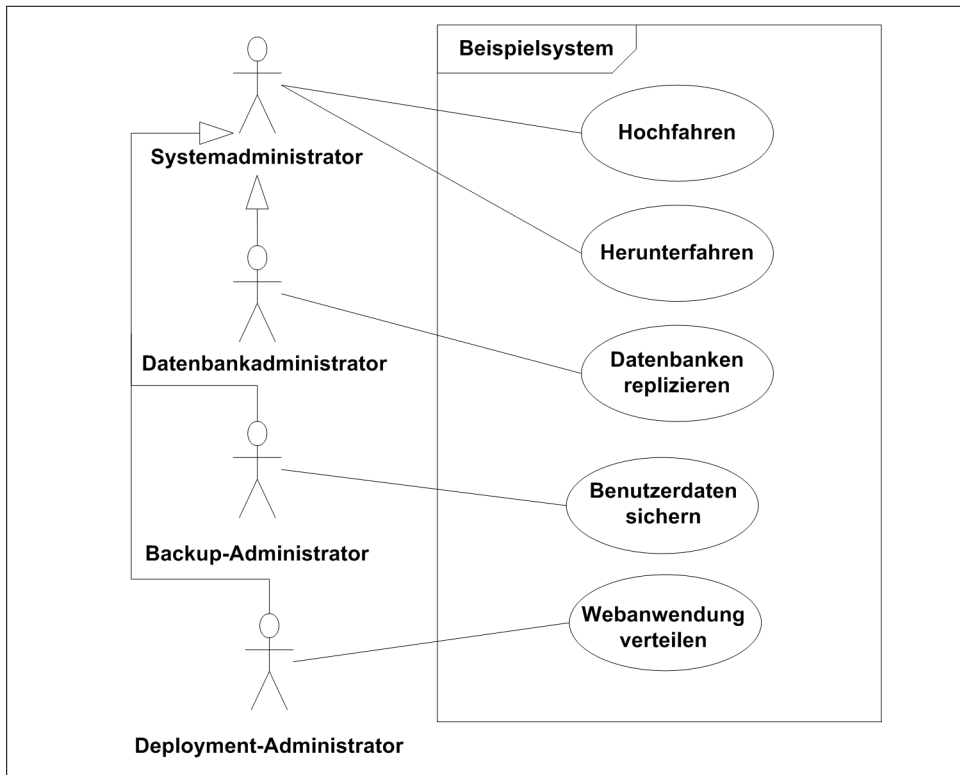


Abbildung 7-11: Akteur-Generalisierung: Der Systemadministrator ist der generische Basisakteur; die anderen drei sind Spezialisierungen.

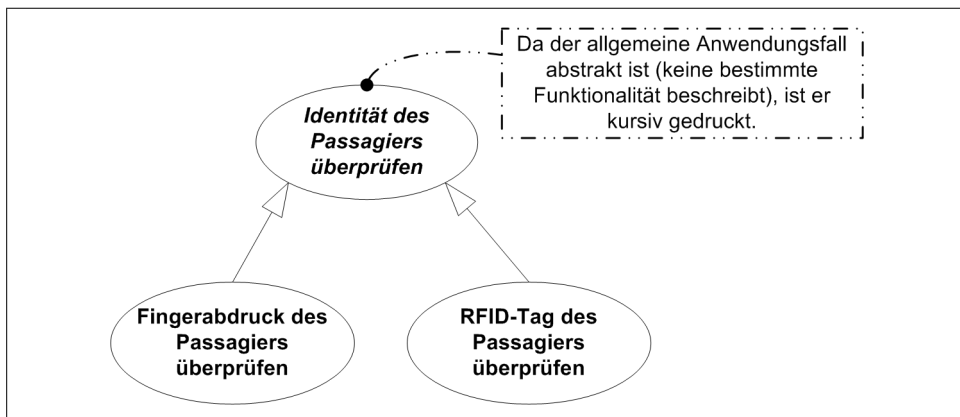


Abbildung 7-12: Anwendungsfallgeneralisierung

Enthaltene Anwendungsfälle

Sie können gemeinsame Funktionalität mehrerer Klassen herausziehen und einen gemeinsam genutzten Anwendungsfall daraus kreieren, der sich einbinden lässt. Anders als Anwendungsfälle, die erweitert werden (siehe nächster Abschnitt), sind Anwendungsfälle, in die andere Anwendungsfälle eingebunden werden, ohne diese nicht komplett. Die enthaltene Funktionalität ist nicht optional; sie wurde lediglich extrahiert, um sie für andere Anwendungsfälle wiederverwendbar zu machen.

Die Einbindung von Anwendungsfällen zeigen Sie mit einer gestrichelten Linie, deren offene Pfeilspitze (Abhängigkeit) von dem Basis-Anwendungsfall auf den eingebundenen Anwendungsfall zeigt und mit dem Schlüsselwort `include` beschriftet ist. Abbildung 7-13 zeigt ein Beispiel.

Include oder Includes?

Zwischen UML-Modellierern herrscht Uneinigkeit in der Frage, ob das Schlüsselwort richtig `include` bzw. `extend` oder `includes` bzw. `extends` lautet. Eigentlich sollte man annehmen, dass die UML-Spezifikation dazu ein klares Wort sagt, aber in der Spezifikation von UML 2.0 heißt es im Abschnitt über Anwendungsfälle `include` und `extend`, aber in den dazugehörigen Beispielen `includes` und `extends`! Man kann wohl getrost davon ausgehen, dass beides richtig ist.

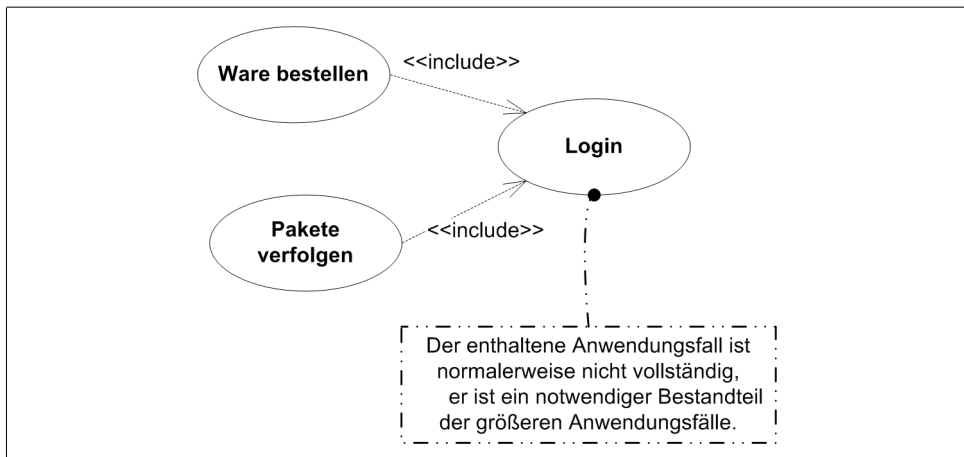


Abbildung 7-13: Einbindung von Anwendungsfällen

Erweiterung von Anwendungsfällen

Die UML bietet die Möglichkeit, einem Anwendungsfall zusätzliche Funktionalität zu verleihen, wenn bestimmte Bedingungen erfüllt sind. Wenn Sie zum Beispiel eine Bankanwendung modellieren, können Sie einen Anwendungsfall namens `Konto eröffnen` einführen, in dem festgelegt wird, wie der Benutzer bei der Bank ein neues Konto eröffnen kann. Sie kön-

nen ein Gemeinschaftskonto anlegen, so dass ein Benutzer noch weitere Personen als Kontoinhaber benennen kann. Die Gemeinschaftskonto-Funktionalität kann mit einem anderen Anwendungsfall namens Gemeinschaftskontoinhaber hinzufügen angelegt werden. In diesem Fall lautet die Bedingung für die Erweiterung, dass mehr als ein Mitglied der Bankanwendung vorhanden ist.

Die UML legt ganz klar fest, dass ein Basis-Anwendungsfall ein kompletter, eigenständiger Anwendungsfall sein muss. Die Erweiterungs-Anwendungsfälle sind im Allgemeinen kleiner in ihrem Geltungsbereich und stellen eine Zusatzfunktionalität dar. Somit sind sie außerhalb des Basis-Anwendungsfalls unter Umständen nicht besonders nützlich.

Jeder Anwendungsfall, den Sie erweitern möchten, muss klar definierte *Erweiterungspunkte* (extension points) haben. Ein Erweiterungspunkt ist ein Punkt in dem Anwendungsfall, an dem der Erweiterungs-Anwendungsfall angedockt werden und seine Funktionalität hinzufügen kann. Die UML kennt keine spezielle Syntax für Erweiterungspunkte; sie sind normalerweise Text in freier Form oder Nummern von Schritten, wenn die Anwendungsfall-Funktionalität als nummerierte Liste dargestellt wird.

Erweiterungspunkte werden im ovalen Anwendungsfallsymbol, oder, wenn sie die Classifier-Notation verwenden, in einem separaten Abschnitt angegeben. Abbildung 7-14 zeigt einen Anwendungsfall mit Erweiterungspunkten.

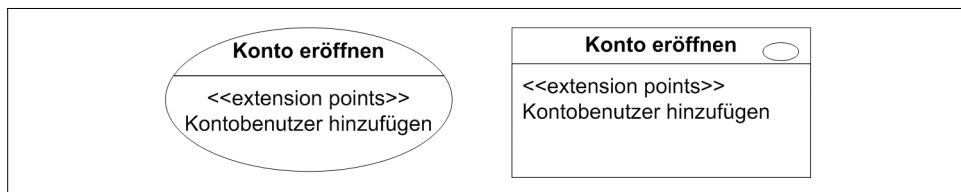


Abbildung 7-14: Oval- und Classifier-Notation für einen Anwendungsfall mit Erweiterungspunkten

Die Erweiterung eines Anwendungsfalls wird mit einer gestrichelten Linie verdeutlicht, deren offene Pfeilspitze (Abhängigkeit) von dem Erweiterungs-Anwendungsfall auf den Basis-Anwendungsfall zeigt und die mit dem Schlüsselwort *extend* beschriftet ist. Abbildung 7-15 zeigt ein Beispiel für die Anwendungsfall-Erweiterung.

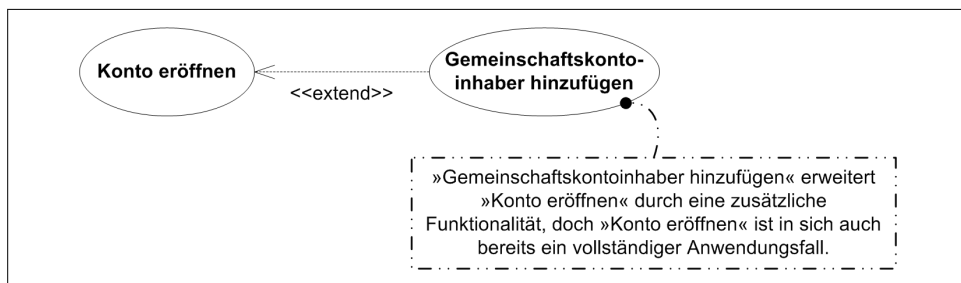


Abbildung 7-15: Anwendungsfall-Erweiterung

Wenn Sie es genauer haben möchten, können Sie angeben, an welcher Stelle des Basis-Anwendungsfalls die neue Funktionalität ansetzt, indem Sie einen Erweiterungspunkt angeben und der Abhängigkeitslinie eine Notiz beifügen. Optional können Sie angeben, unter

welcher Bedingung die Erweiterung ausgeführt wird, etwa wie `bewerber > 1`. Abbildung 7-16 zeigt die Erweiterung eines Anwendungsfalls mit einer Notiz, auf der der Erweiterungspunkt und die Ausführungsbedingung für die Zusatzfunktionalität vermerkt sind.

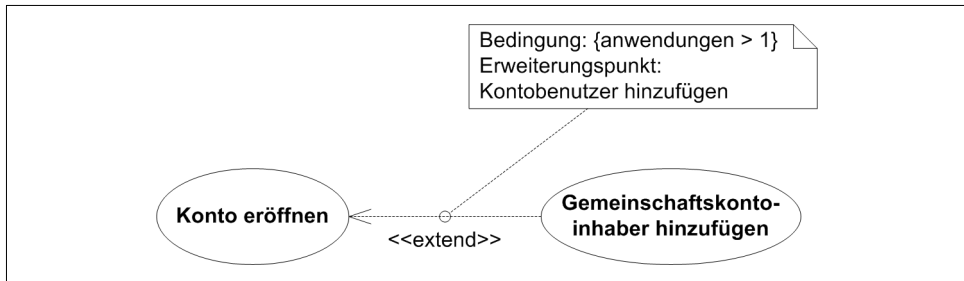


Abbildung 7-16: Anwendungsfall-Erweiterung mit einer Notiz, die Bedingungen anzeigt

Wenn das System einen Erweiterungspunkt in einem Anwendungsfall findet, werden die Bedingungen ausgewertet, die mit diesem Anwendungsfall zusammenhängen. Ist eine Bedingung erfüllt, wird die zugehörige Erweiterungsfunktionalität ausgeführt. Sobald alle passenden Erweiterungs-Anwendungsfälle ausgeführt wurden, geht es mit dem nächsten Schritt des ursprünglichen Ablaufs im Basis-Anwendungsfall weiter.

Umfang von Anwendungsfällen

Wie bereits gesagt, ist ein Anwendungsfall ein eigenes Stück Funktionalität, also feinkörnig genug, dass der Benutzer damit sein gewünschtes Ziel erreichen kann. Es ist eine Kunst, den richtigen Umfang für einen Anwendungsfall festzulegen, aber die UML stellt mehrere Anforderungen, die Ihnen die Entscheidungen erleichtern:

- Ein Anwendungsfall muss von einem Akteur instanziiert werden.
- Wenn ein Anwendungsfall zum Ende gekommen ist, gibt es keine weiteren Ein- und Ausgaben. Entweder wurde die gewünschte Funktionalität geboten oder es trat ein Fehler auf.
- Nachdem ein Anwendungsfall zum Ende gekommen ist, befindet sich das System entweder in einem Zustand, in dem er neu gestartet werden kann oder in einem Fehlerzustand.

Eine beliebte Faustregel lautet: Fragen Sie sich nach Abschluss des Anwendungsfalls, ob Sie jetzt essen gehen könnten. Wenn ja, so bedeutet das, dass ein vernünftig zugeschnittenes Ziel von dem Initiator erreicht wurde. So ist beispielsweise In den Warenkorb nicht das eigentliche Ziel, das der Benutzer verfolgt; Kaufen ist da schon weit besser. Zwar gehört auch das Legen von Waren in den Warenkorb zum Kaufen, aber es kommt noch einiges mehr an Funktionalität hinzu, wie etwa das Anmelden, Eingeben von Rechnungs- und Versanddaten und Bestätigen der Bestellung.

Da Anwendungsfälle vor allem dazu da sind, gewünschte Funktionalität zu verdeutlichen, kann der Umfang eines Anwendungsfalls je nach Zielgruppe und Modellierungszweck sehr unterschiedlich sein.