12 Anwendungsfalldiagramm

In Abschnitt 1.5 haben Sie gelesen, dass Anwendungssysteme in einen Unternehmenskontext eingebunden sind und dort Geschäftsprozesse unterstützen. Aus diesem Blickwinkel ist die Beschreibung der funktionalen Anforderungen, also dessen, was ein Anwendungssystem seinen Benutzern anbietet oder anbieten soll, besonders wichtig. Diese Beschreibung umfasst z. B. Funktionalitäten, Abläufe und Ausnahmebehandlungen, zu beachtende Geschäftsregeln sowie externe Schnittstellen etwa zu Datenbanken und anderen Anwendungssystemen.

Hierfür bietet die UML die Konzepte *Akteur* und *Anwendungsfall* an, die in den beiden nächsten Abschnitten vorgestellt werden.

12.1 Akteure

Innerhalb von Geschäftsprozessen kommunizieren menschliche oder maschinelle »Benutzer« mit dem zu entwickelnden Anwendungssystem, um bestimmte Aufgaben durchzuführen und konkrete Ziele zu erreichen. Im Normalfall wird es mehrere solcher *Benutzer* geben, die in Bezug auf das Anwendungssystem eine bestimmte *Rolle* spielen. Mit jeder dieser Rollen sind bestimmte Aufgaben verknüpft.

Ähnlich wie eine Klasse eine Abstraktion konkreter Objekte darstellt, abstrahiert ein *Akteur* (*actor*) eine bestimmte Rolle konkreter Benutzer der realen Welt. Ein konkreter Benutzer kann dabei durchaus mehrere Rollen spielen und daher durch mehrere Akteure modelliert werden. Akteure modellieren also Typen von Benutzern oder Systemen, welche außerhalb des zu entwickelnden Anwendungssystems existieren und mit diesem interagieren, also mit ihm auf irgendeine Art und Weise Informationen austauschen.

In UML-Diagrammen werden Akteure als »Strichmännchen«  dargestellt, wobei der Name des Akteurs in Fettdruck unter die Figur geschrieben wird. Eigene Stereotype mit entsprechenden Sinnbildern z. B. für Akteure, die maschinelle Benutzer (externe Systeme) verkörpern, können die Lesbarkeit der Diagramme erhöhen.

Beispiel 12–1 Im Fallbeispiel »Bestellwesen« aus Kapitel 8 verwalten Angestellte des Warenhauses als Sachbearbeiter die Daten der Privat- bzw. Firmenkunden, führen Bestellungen durch und veranlassen den Ausdruck von Rechnungen und Mahnungen. Es ergeben sich also die beiden in Abbildung 12–1 dargestellten (menschlichen) Akteure Privatkunden-Sachbearbeiter und Firmenkunden-Sachbearbeiter sowie die (maschinellen) Akteur Drucker und Systemuhr.

Abb. 12–1 *Akteure im Bestellwesen*

Unterschiedliche Akteure können gemeinsame Eigenschaften haben und somit – ebenso wie unterschiedliche Klassen – in einer Generalisierungsbeziehung zueinander stehen. Analog zur Generalisierung von Klassen können die spezielleren Akteure mindestens die Rollen der allgemeineren Akteure spielen.

Beispiel 12–2 Im Fallbeispiel »Bestellwesen« haben die beiden Akteure Privatkunden-Sachbearbeiter und Firmenkunden-Sachbearbeiter gemeinsame Verantwortlichkeiten wie z.B. »Bestellung erfassen« und »Bestellposten ermitteln«. Zur Redundanzvermeidung und besseren Übersichtlichkeit kann man diese in einen allgemeineren Akteur Sachbearbeiter zusammenfassen, wodurch die in Abbildung 12–2 gezeigte Akteur-Generalisierungshierarchie entsteht.

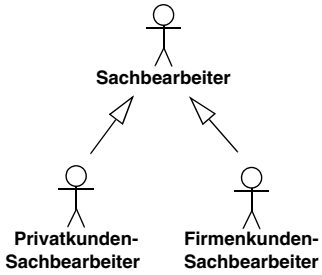

Abb. 12–2 *Akteur-Generalisierungshierarchie im Bestellwesen*

12.2 Anwendungsfälle

Funktionen des Anwendungssystems, die von den Akteuren zur Durchführung ihrer Aufgaben benötigt werden, modelliert man als so genannte Anwendungsfälle. Jeder *Anwendungsfall* (*use case*) formuliert eine in sich abgeschlossene Teilfunktionalität des Anwendungssystems, die für mindestens einen Akteur ein bestimmtes Ergebnis innerhalb des Geschäftsprozesses erbringt. Darüber hinaus können weitere Akteure an der Durchführung des Anwendungsfalls beteiligt sein.

Die Kommunikation der Akteure mit dem System wird durch Assoziationen zwischen Akteuren und Anwendungsfällen modelliert, wobei die Richtung des Informationsflusses durch entsprechende Navigierbarkeit der Assoziationen modelliert werden kann.

Grafisch werden Anwendungsfälle im *Anwendungsfalldiagramm* als Ellipsen ⬭ dargestellt, in denen der Name des jeweiligen Anwendungsfalls in Fettdruck aufgeführt

wird. Die Anwendungsfälle können optional in einem Rechteck gebündelt werden, das die »Systemgrenze« des Anwendungssystems repräsentiert. Assoziationen zwischen Anwendungsfällen und Akteuren werden – ebenso wie Assoziationen im Klassendiagramm – als durchgezogene Linien gezeichnet, wobei offene Pfeilspitzen ggf. die Richtung des Informationsflusses angeben (vgl. Abschnitt 9.4).

Beispiel 12–3 Ein Anwendungsfall im Bestellwesen ist das Erfassen einer Bestellung. Hierbei wählt der Sachbearbeiter mit dem Anwendungssystem zunächst den Kunden und dann die einzelnen Produkte und Bestellmengen aus. Abbildung 12–3 zeigt den entsprechenden Anwendungsfall »Bestellung erfassen« zusammen mit dem Akteur Sachbearbeiter und der einen Informationsfluss in beiden Richtungen darstellenden (Kommunikations-)Assoziation in der UML-Notation.

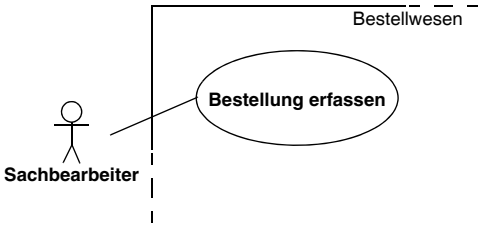

Abb. 12–3 *Assoziation zwischen Akteur und Anwendungsfall im Bestellwesen*

Anwendungsfälle spielen eine zentrale Rolle bei der Kommunikation zwischen Anforderungsermittler (Analytiker) und dem Benutzer. Textuell werden sie daher zunächst mehr oder weniger »prosaisch« aus der Sicht der Akteure beschrieben, wobei das Anwendungssystem als »Black-Box« betrachtet wird. Im Zuge der fortschreitenden Präzisierung der Beschreibung können – analog zu Operationsspezifikationen – Vor- und/oder Nachbedingungen ergänzt werden. Die *Vorbedingung* (precondition) eines Anwendungsfalls grenzt die Situationen ein, in denen der Anwendungsfall ausgeführt werden kann bzw. darf; sie muss vom System überprüfbar sein. Die *Nachbedingung* (postcondition) präzisiert das für den Akteur relevante vom System zu erbringende Ergebnis des Anwendungsfalls.

Jeder Anwendungsfall formuliert die möglichen Abläufe und Interaktionen, die zwischen Akteuren und dem Anwendungssystem im Rahmen der Teilfunktionalität stattfinden. Um die Beschreibung des »normalen« Ablaufs (*main flow*) eines Anwendungsfalls möglichst einfach zu halten, werden mögliche Abweichungen vom normalen Ablauf in eigenen Abschnitten aufgeführt. Bei den Abweichungen unterscheidet man zwei unterschiedliche Varianten. Bei so genannten *alternativen Abläufen* (*alternative flows*) wird der Anwendungsfall wie beim normalen Ablauf erfolgreich beendet, und es ist auch hier die Nachbedingung des normalen Ablaufs erfüllt. Anders verhält es sich bei *Ausnahmesituationen* (*exceptional flows*). Hier ist der erfolgreiche Ablauf des Anwendungsfalls gefährdet und es wird jeweils eine gesonderte Nachbedingung angegeben.

Zur einheitlichen textuellen Spezifikation der Anwendungsfälle empfiehlt es sich, organisations- oder zumindest projektweit ein verbindliches Schema wie z.B. das in Abbildung 12–4 gezeigte festzulegen. In der UML selbst ist ein solches Schema nicht vorgegeben.

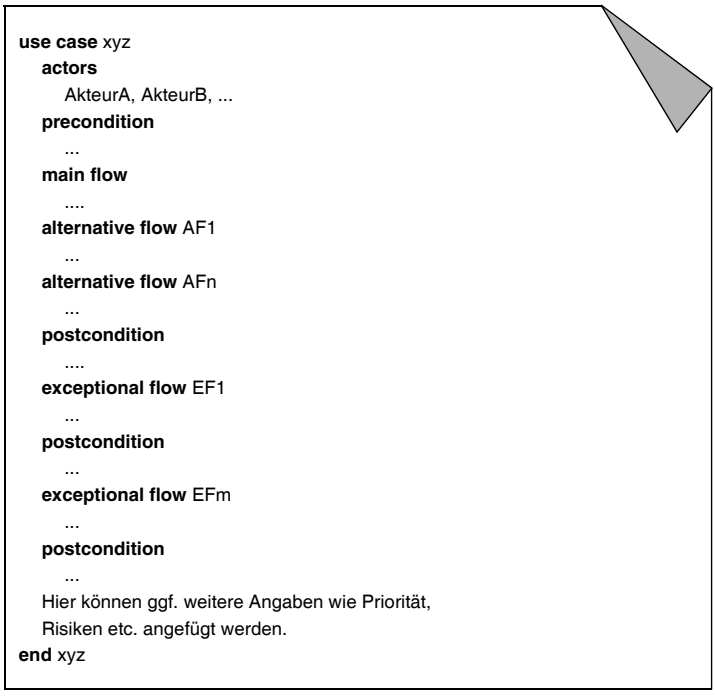

Abb. 12–4 *Schema für die textuelle Spezifikation eines Anwendungsfalls*

Beispiel 12–4 Im Bestellwesen müssen Kundendaten nach Bezahlung der letzten offenen Rechnung entsprechend einer gesetzlich vorgeschriebenen Aufbewahrungsfrist gespeichert bleiben. Sollen die Daten eines Kunden gelöscht werden, markiert der Sachbearbeiter sie daher zunächst lediglich als »gelöscht« (Deaktivieren). Erst nach Ablauf der Aufbewahrungsfrist werden sie vom Anwendungssystem automatisch gelöscht. Die textuelle Spezifikation des entsprechenden Anwendungsfalls »Kunde deaktivieren« zeigt Abbildung 12–5.

use case Kunde deaktivieren

actors

Sachbearbeiter

precondition

-

main flow

Der Sachbearbeiter bestimmt den zu deaktivierenden Kunden anhand der Kundennummer.
Die Deaktivierung muss vom Sachbearbeiter bestätigt werden.

alternative flow Kundennummer unbekannt

Der Sachbearbeiter wählt zunächst den zu deaktivierenden Kunden aus einer Liste aller Kunden aus und deaktiviert ihn dann.

postcondition

Der Kunde ist deaktiviert.

exceptional flow Offene Rechnung

Hat der ausgewählte Kunde noch Rechnungen offen, so darf er nicht deaktiviert werden.

postcondition

Keine Deaktivierung durchgeführt.

exceptional flow Falsche Kundennummer

Zur angegebenen Kundennummer existiert kein Kunde, so dass auch keine Deaktivierung durchgeführt werden kann.

postcondition

Keine Deaktivierung durchgeführt.

end Kunde deaktivieren

Abb. 12–5 *Textuelle Spezifikation des Anwendungsfalls »Kunde deaktivieren«*

Als weiteres Beispiel zeigt Abbildung 12–6 die textuelle Beschreibung des Anwendungsfalls »Bestellung erfassen«. Hierbei sind die einzelnen Schritte der Ablaufbeschreibungen zur besseren Übersichtlichkeit nummeriert.

use case Bestellung erfassen

actors

Sachbearbeiter

precondition

Es existieren Produkte

main flow

1. Der Sachbearbeiter nimmt die Bestellung (per Telefon, Fax etc. entgegen.
2. Er identifiziert den Kunden im System.
3. Er nimmt alle Bestellposten mit den jeweiligen Produkten und Bestellmengen auf.

alternative flow Kundendaten nicht aktuell

2.a Stellt der Sachbearbeiter bei Aufnahme der Bestellung fest, dass es sich um einen Neukunden handelt oder dass sich die Kundendaten seit der letzten Bestellung geändert haben, werden die Kundendaten zusätzlich erfasst bzw. aktualisiert.

alternative flow Vorauszahlung fordern

3.a Gibt es für einen Privatkunden noch offene Rechnungen, markiert der Sachbearbeiter die Bestellung als »im Voraus zu bezahlen« und veranlasst, dass zusätzlich die Rechnung ausgedruckt und an den Kunden gesendet wird.

alternative flow Kreditrahmen verhandeln

3.b Hat ein Firmenkunde noch offene Rechnungen und übersteigt deren Volumen zuzüglich des Wertes der aktuellen Bestellung den Kreditrahmen des Kunden um maximal 10.000 Euro, wird zusätzlich zur Vorauszahlung über eine entsprechende Erweiterung des Kreditrahmens verhandelt.

postcondition

Die Bestellung ist erfasst. Die Kundendaten sind aktuell. Gegebenenfalls ist der Kunde neu erfasst, die Bestellung als »im Voraus zu bezahlen« markiert und eine Verhandlung über den Kreditrahmen aufgenommen.

exceptional flow Bestellwert Firmenkunde zu hoch

3.c Der Kreditrahmen eines Firmenkunden müsste um mehr als 10.000 Euro erweitert werden, wozu nur der Abteilungsleiter berechtigt ist. Der Abteilungsleiter wird vom Sachbearbeiter entsprechend informiert und der Bestellvorgang als »Wiedervorlage« markiert.

postcondition

Die Bestellung ist erfasst und als »Wiedervorlage« markiert. Die Kundendaten sind aktuell. Der Abteilungsleiter ist informiert.

end Bestellung erfassen

Abb. 12–6 *Textuelle Spezifikation des Anwendungsfalls »Bestellung erfassen«*

Jeder Anwendungsfall stellt eine »Familie« von möglichen konkreten Abläufen dar. Bei der »Instanziierung« eines Anwendungsfalls wird seine allgemein beschriebene Funktionalität für einen speziellen, vorgegebenen Zustand des Anwendungssystems konkret präzisiert. Eine solche Instanziierung, d.h. ein konkreter Ablauf eines Anwendungsfalls, wird als *Szenario* bezeichnet.

Beispiel 12–5 Ein typisches Szenario des Anwendungsfalls »Bestellung erfassen« ist die Annahme einer von der Privatkundin Karoline Meier aufgegebenen Bestellung über zwanzig Kugelschreiber und zehn Schreibblöcke für den Fall, dass die Kundin nicht im Zahlungsverzug ist und alle Produkte vorhanden sind:

Szenario Bestellung erfassen – Alle Produkte vorhanden

Nach der Entgegennahme der Bestellung wählt der Sachbearbeiter im Bestellwesen-Anwendungssystem zunächst die Daten der Kundin Karoline Meier aus und gibt dann die Bestellposten mit den Produkten Kugelschreiber und Schreibblock sowie den Mengen zwanzig und zehn ein. Anschließend prüft der Sachbearbeiter, ob die Kundin in Zahlungsverzug ist. Da dies nicht der Fall ist, markiert er die Bestellung nicht als »Im Voraus zu bezahlen« und es wird keine Vorabrechnung ausgedruckt und versendet.

end Bestellung erfassen – Alle Produkte vorhanden

Abb. 12–7 *Szenario im Bestellwesen*

12.3 Beziehungen zwischen Anwendungsfällen

Bei der Anforderungsermittlung für große Anwendungssysteme können Fälle eintreten, die eine weitere Strukturierung der Anwendungsfälle notwendig machen. Einerseits kann die Vielzahl der Anwendungsfälle die Anforderungsspezifikation unüberschaubar machen, andererseits können einzelne Anwendungsfälle teilweise sehr komplex werden. Oft beinhalten mehrere Anwendungsfälle ähnliche oder gleiche Teilfunktionalitäten, die nur an einer Stelle spezifiziert werden sollten, um Redundanz zu vermeiden. Letztendlich möchte man manchmal bestimmte Funktionalitäten bewusst offen oder zumindest flexibel lassen, um besser für zukünftige Änderungen oder Erweiterungen gewappnet zu sein.

Im Fall vieler kleiner, inhaltlich zusammengehöriger Anwendungsfälle können diese in Paketen zusammengefasst werden. Dies verbessert zwar den Überblick, hilft aber nicht, redundante Beschreibungen zu vermeiden, komplexe Anwendungsfälle zu strukturieren und Abhängigkeiten zwischen verschiedenen Anwendungsfällen darzustellen. Als Lösung bietet die UML die include- und die extend-Beziehung sowie die Generalisierung zwischen Anwendungsfällen an.

Include

Die include-Beziehung wird verwendet, wenn verschiedene Anwendungsfälle dieselbe Teilfunktionalität beinhalten. Diese Teilfunktionalität wird zur Redundanzvermeidung in einem separaten Anwendungsfall beschrieben, der von den »benutzenden« Anwendungsfällen (Basisanwendungsfälle) verwendet wird. Eine include-Beziehung von einem Basisanwendungsfall *A* zu einem Anwendungsfall *B* bedeutet also, dass *A* die in *B* spezifizierte Teilfunktionalität benutzt und somit von dessen erfolgreichem Ablauf bzw. seinem Ergebnis abhängig ist.

In der textuellen Spezifikation des Basisanwendungsfalls wird die include-Beziehung durch das Schlüsselwort include gefolgt vom Namen des benutzten Anwendungsfalls dargestellt (vgl. Abb. 12–10).

Im Anwendungsfalldiagramm wird eine include-Beziehung zwischen zwei Anwendungsfällen durch einen gestrichelten Pfeil mit offener Spitze - - - ⩾ dargestellt[1], der

die Symbole der Anwendungsfälle verbindet. Der Pfeil ist vom benutzenden Anwendungsfall zum benutzten Anwendungsfall gerichtet und wird mit «include» bezeichnet.

Beispiel 12–6 Bei näherer Untersuchung des Anwendungsfalls »Bestellung erfassen« (vgl. Abb. 12–6) stellt sich die Ermittlung der einzelnen Posten einer Bestellung als komplexer Vorgang dar, dessen präzisere Beschreibung die textuelle Spezifikation stark verkomplizieren würde. Besser ist es, einen eigenen Anwendungsfall »Bestellposten ermitteln« anzulegen, der mittels einer include-Beziehung vom (Basis-)Anwendungsfall »Bestellung erfassen« benutzt wird (Abb. 12–8).

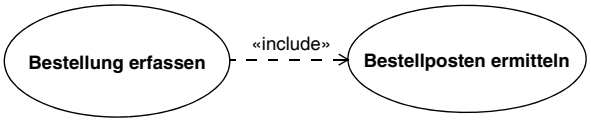

Abb. 12–8 *Visualisierung der* include*-Beziehung zwischen zwei Anwendungsfällen*

Extend

Eine extend-Beziehung von einem Anwendungsfall *B* zu einem Anwendungsfall *A* bedeutet, dass Anwendungsfall *A* unter bestimmten Bedingungen durch die im Anwendungsfall *B* beschriebene Funktionalität erweitert werden kann. In anderen Worten: Die Funktionalität des Anwendungsfalls *B* ist eine optionale Erweiterung des Anwendungsfalls *A*.

In der textuellen Spezifikation von Anwendungsfall *A* (Basisanwendungsfall) wird die Stelle, an der die Funktionalität eines erweiternden Anwendungsfalls eingefügt werden kann, als *Erweiterungspunkt* (*extension point*) gekennzeichnet (vgl. Abb. 12–10). In der Spezifikation der extend-Beziehung werden dann sowohl der entsprechende Erweiterungspunkt als auch die Bedingung angegeben, bei deren Eintreten der Anwendungsfall *B* (Erweiterung) einzufügen ist.

Beispiel 12–7 In der textuellen Spezifikation des Anwendungsfalls »Bestellung erfassen« in Abbildung 12–6 stellen die beiden alternativen Abläufe »Vorauszahlung fordern« und »Kreditrahmen verhandeln« Funktionalitäten dar, die nur unter bestimmten Bedingungen zur Anwendung kommen und eher unerheblich zur Basisfunktionalität »Erfassen der Daten einer Bestellung« beitragen. Beide kommen dann zur Anwendung, wenn der Sachbearbeiter nach der Aufnahme einer Bestellung feststellt, dass es für den Kunden noch offene Rechnungen gibt, seine Bonität also schlecht ist. Zur übersichtlicheren Spezifikation werden diese Varianten in eigene Anwendungsfälle ausgelagert und über extend-Beziehungen mit dem Basisanwendungsfall verknüpft.

1. Include- (und Extend-)Beziehungen werden im Anwendungsfalldiagramm wie Abhängigkeiten (vgl. Abschnitt 10.3) dargestellt, obwohl ihre Semantik eher der einseitig navigierbarer Assoziationen (vgl. Abschnitt 9.4) entspricht. Die UML verbietet jedoch Assoziationen zwischen Anwendungsfällen des gleichen Systems.

Abbildung 12–9 beinhaltet die textuellen Spezifikationen der Anwendungsfälle »Bestellung erfassen« und »Vorauszahlung fordern«, Abbildung 12–10 die der entsprechenden extend-Beziehung. Da die Nachbedingung des Anwendungsfalls »Bestellung erfassen« auch von der Variante (Kundendaten nicht aktuell) erfüllt wird, ist für diese keine gesonderte Nachbedingung angegeben. Des Weiteren werden die Teilaufgaben »Kunde auswählen« und »Kunde bearbeiten« als separate Anwendungsfälle modelliert und – ebenso wie der Anwendungsfall »Bestellposten ermitteln« – über include-Beziehungen mit dem Anwendungsfall »Bestellung erfassen« verbunden.

use case Bestellung erfassen

actors
Sachbearbeiter

precondition
-

main flow
Der Sachbearbeiter nimmt die Bestellung (per Telefon, Fax, Brief etc.) entgegen, identifiziert den Kunden im System (include »Kunde auswählen«) und nimmt dann alle Bestellposten mit den jeweiligen Produkten und Bestellmengen auf (include »Bestellposten ermitteln«). Bei schlechter Bonität kann die Bestellung noch an die spezielle Situation angepasst werden (extension point: Anpassen).

alternative flow Kundendaten nicht aktuell
Stellt der Sachbearbeiter bei Aufnahme der Bestellung fest, dass es sich um einen Neukunden handelt oder dass sich die Kundendaten seit der letzten Bestellung geändert haben, werden diese zusätzlich erfasst bzw. aktualisiert (include »Kunde bearbeiten«).

postcondition
Die Bestellung ist erfasst. Der Kunde ist ggf. neu erfasst, die Kundendaten sind aktuell.

exceptional flow Bestellwert Firmenkunde zu hoch
Der Kreditrahmen eines Firmenkunden müsste um mehr als 10.000 Euro erweitert werden, wozu nur der Abteilungsleiter berechtigt ist. Der Abteilungsleiter wird vom Sachbearbeiter entsprechend informiert und der Bestellvorgang als »Wiedervorlage« markiert.

postcondition
Die Bestellung ist erfasst und als »Wiedervorlage« markiert. Die Kundendaten sind aktuell.
Der Abteilungsleiter ist informiert.

end Bestellung erfassen

---

use case Vorauszahlung fordern

actors
Sachbearbeiter

precondition
Der Kunde ist erfasst, die Bestellung ist erfasst.

main flow
Der Sachbearbeiter markiert die Bestellung als »im Voraus zu bezahlen« und veranlasst, dass die Rechnung ausgedruckt und an den Kunden gesendet wird.

postcondition
Die Bestellung ist als »im Voraus zu bezahlen« markiert, der Kunde erhält die Rechnung vorab.

end Vorauszahlung fordern

Abb. 12–9 *Textuelle Spezifikationen der Anwendungsfälle »Bestellung erfassen« und »Vorauszahlung fordern«*

extend relationship

base

»Bestellung erfassen«

extensionPoint

Anpassen

extension

»Vorauszahlung fordern«

condition

Bonität schlecht, d.h., es gibt offene Rechnungen des Kunden und, falls es sich um einen Firmenkunden handelt, der Kreditrahmen ist ausgeschöpft.

end

Abb. 12–10 *Textuelle Spezifikationen einer* extend*-Beziehung zwischen den Anwendungsfällen »Bestellung erfassen« und »Vorauszahlung fordern«*

Zur Darstellung der Erweiterungspunkte eines Basisanwendungsfalls im Anwendungsfalldiagramm wird das Anwendungsfallsymbol horizontal mit einer durchgezogenen Linie unterteilt. Über dieser Linie steht der Name des Anwendungsfalls, unter ihr werden seine Erweiterungspunkte aufgelistet. Die extend-Beziehung wird im Anwendungsfalldiagramm ebenso wie die include-Beziehung als gestrichelter Pfeil mit offener Spitze - - - ⟩ dargestellt. Der Pfeil ist vom Anwendungsfall, der die optionale Erweiterung darstellt (Anwendungsfall *B*), zum Basisanwendungsfall (Anwendungsfall *A*) gerichtet[1] und mit «extend» gekennzeichnet. Optional kann eine an den Pfeil geheftete Notiz die Erweiterungspunkte und ggf. die Bedingung, unter der die Erweiterung eintritt, angeben.

Beispiel 12–8 Als Beispiel für die Darstellung von extend-Beziehungen dienen wieder die Anwendungsfälle »Bestellung erfassen«, »Vorauszahlung fordern« und »Kreditrahmen verhandeln« im Bestellwesen. Der Anwendungsfall »Bestellung erfassen« beinhaltet den Erweiterungspunkt Anpassen, da im Falle schlechter Bonität eines Kunden eine Vorauszahlung gefordert werden kann. Bei Firmenkunden wird in solchen Fällen zusätzlich der Kreditrahmen neu verhandelt. Zwischen dem Basisanwendungsfall »Bestellung erfassen« und den Anwendungsfällen »Vorauszahlung fordern« und »Kreditrahmen verhandeln« werden daher extend-Beziehungen modelliert (Abb. 12–11). In den Notizen zu den extend-Beziehungen sind zusätzlich der betroffene Erweiterungspunkt (Anpassen) des Basisanwendungsfalls »Bestellung erfassen« und die Bedingungen angegeben, unter denen die Erweiterungen ausgeführt werden. Man erkennt hieran, dass verschiedene extend-Beziehungen an denselben Erweiterungspunkt angeknüpft werden können.

1. Die Richtung drückt aus, welcher der beiden Anwendungsfälle »Kenntnis« vom anderen hat. Daher zeigt die Pfeilspitze bei der include-Beziehung genau entgegengesetzt, also vom Basisanwendungsfall zum benutzten Anwendungsfall hin.

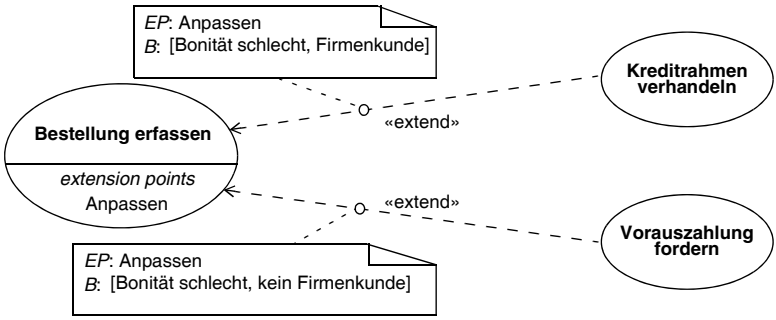

Abb. 12–11 *Visualisierung der* extend*-Beziehung zwischen Anwendungsfällen*

Sowohl die include- als auch die extend-Beziehung fügen zusätzliche Funktionalität in eine Basisfunktionalität ein, lassen aber keine Aussage über die zeitliche Reihenfolge ihrer Ausführung zu. Im Falle der include-Beziehung komplettiert die zusätzliche Funktionalität die des Basisanwendungsfalls dann, wenn dieser bei seinem Ablauf die include-Beziehung erreicht – dies kann zu Beginn, am Ende oder irgendwann während seines Ablaufs passieren, manchmal auch überhaupt nicht. Im Falle der extend-Beziehung ist zusätzliche Funktionalität für die des Basisanwendungsfalls optional. Sie wird bei Erreichen des Erweiterungspunktes ja nur dann aktiviert, wenn die Bedingung der extend-Beziehung zutrifft, so dass der Basisanwendungsfall also auch ohne die zusätzliche Funktionalität ein sinnvolles Ergebnis liefern muss.

Man kann sich die include-Beziehung auch als Operationsaufruf vorstellen, bei dem der Basisanwendungsfall den benutzten Anwendungsfall kennt und den Rückgabewert benötigt, während eine extend-Beziehung eher einem Beobachter des Basisanwendungsfalls entspricht, der bei Erreichen eines Erweiterungspunktes und zutreffender Bedingung den erweiternden Anwendungsfall aktiviert.

Die include-Beziehung vermeidet Redundanz und erlaubt die Wiederverwendung von Teilfunktionalitäten, während die extend-Beziehung die Flexibilität und einfache Erweiterbarkeit des Anwendungsfallmodells sicherstellen soll. In der UML bleibt die präzise Semantik der extend-Beziehung etwas unklar, in jedem Fall ist ihre Beschreibung unangemessen kompliziert. Sie wird daher oft im Sinne einer mit einer Bedingung versehenen include-Beziehung verwendet.

Wichtig ist, bei der Verwendung von include- und extend-Beziehungen zyklische Abhängigkeiten zu vermeiden. Ein Anwendungsfall darf sich selbst also weder direkt noch über mehrere Stufen benutzen oder erweitern.

Generalisierungsbeziehungen

Neben der include- und der extend-Beziehung können Anwendungsfälle auch in einer *Generalisierungsbeziehung* zueinander stehen, wobei der speziellere Anwendungsfall die Aufgabe des allgemeineren Anwendungsfalls umfasst, aber einige Teilaufgaben auf eine spezielle Art und Weise bearbeitet. Der speziellere Anwendungsfall kann also mindestens überall dort Anwendung finden, wo auch der allgemeinere Anwendungsfall

angewendet wird. Bei der Generalisierungsbeziehung werden auch die Beziehungen zwischen Akteuren und dem allgemeineren Anwendungsfall an die spezielleren Anwendungsfälle »vererbt«[1].

Generalisierungsbeziehungen werden im Anwendungsfalldiagramm – analog zum Klassendiagramm – durch Pfeile mit geschlossener, nicht ausgefüllter Spitze ⟶▷ von spezielleren zu allgemeineren Anwendungsfällen hin visualisiert.

Beispiel 12–9 Im Bestellwesen wird die allgemeine Aufgabe »Forderung stellen« durch die spezielleren Aufgaben »Monatsrechnung erstellen«, »Mahnung erstellen« und »Kreditkarte belasten« konkretisiert. Die Kommunikationsbeziehung zum Akteur Sachbearbeiter wird an die spezielleren Anwendungsfälle »vererbt«.

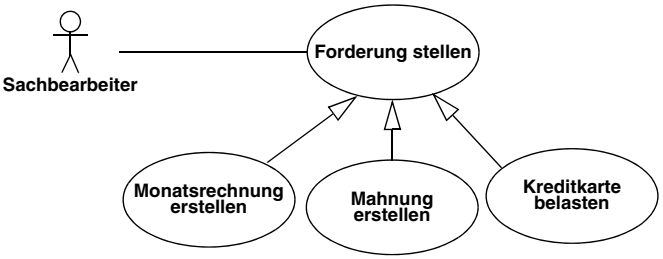

Abb. 12–12 *Visualisierung von Generalisierungsbeziehungen im Anwendungsfalldiagramm*

Beispiel 12–10 Einen größeren Ausschnitt des Anwendungsfalldiagramms für das Bestellwesen zeigt Abbildung 12–13. Es beinhaltet den Akteur Sachbearbeiter und die spezielleren Akteure Firmenkunden-Sachbearbeiter und Privatkunden-Sachbearbeiter sowie den Akteur Drucker. Kunden interagieren nicht direkt mit dem System und sind deshalb keine Akteure.

Im Anwendungsfall »Kunde bearbeiten« muss zunächst der Kunde ausgewählt werden, dessen Angaben zu bearbeiten sind. Da die Auswahl eines Kunden offensichtlich auch für andere Funktionalitäten sinnvoll ist, drängt sich die Modellierung eines eigenen Anwendungsfalls »Kunde auswählen« auf, der zu dem obigen Anwendungsfall in einer include-Beziehung steht. Der Anwendungsfall »Vorauszahlung fordern« stellt einen Spezialfall des Anwendungsfalls »Rechnung erstellen« dar und steht mit diesem in einer Generalisierungsbeziehung. Ebenso stellen die Anwendungsfälle »Rechnung erstellen«, »Monatsrechnung erstellen«, »Mahnung erstellen« und »Kreditkarte belasten« Spezialisierungen des Anwendungsfalls »Forderung stellen« dar.

1. Im Fall der include- bzw. extend-Beziehung schweigt sich die UML über die mit dem benutzten bzw. erweiternden Anwendungsfall kommunizierenden Akteure aus. Es erscheint sinnvoll, dass die Akteure des Basisanwendungsfalls – soweit nichts anderes angegeben ist – zumindest bei der include-Beziehung auch mit den benutzten Anwendungsfällen kommunizieren können. Separate Kommunikationsbeziehungen weisen dann darauf hin, dass Akteure die benutzten Anwendungsfälle auch losgelöst von den Basisanwendungsfällen aktivieren können.

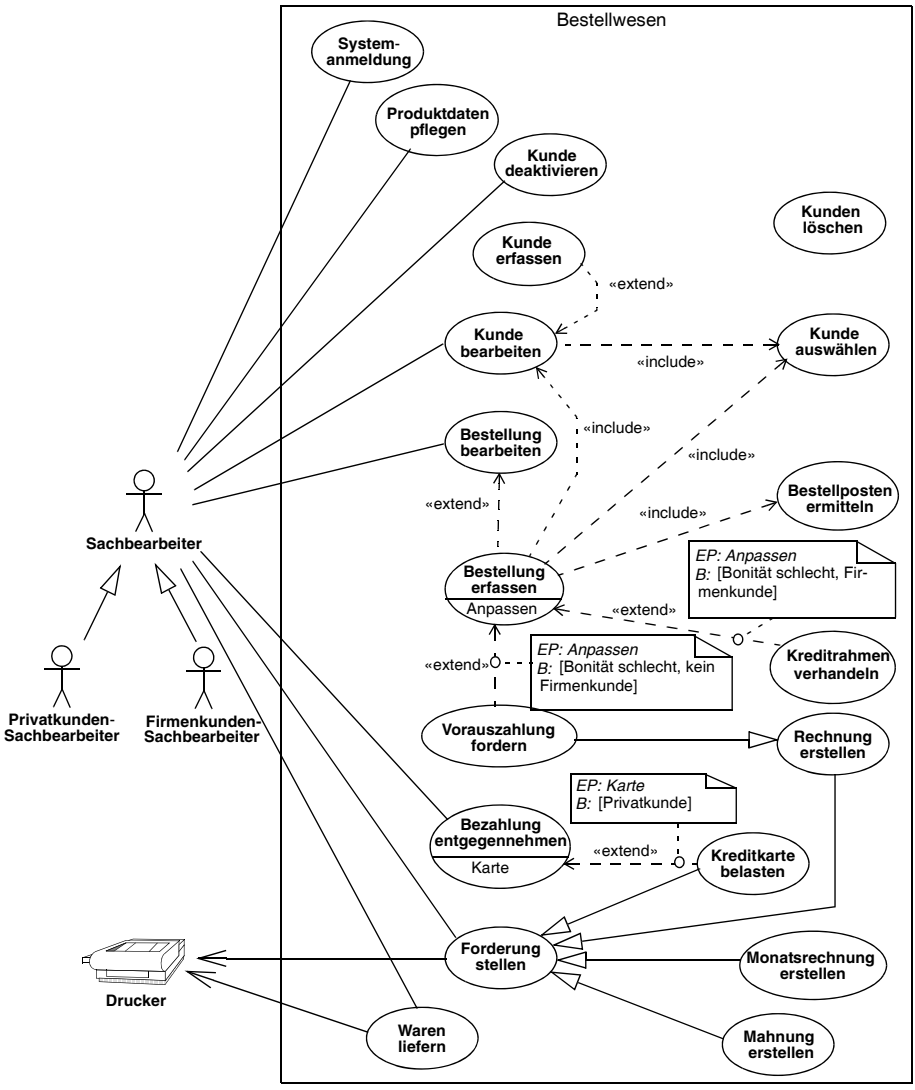

Abb. 12–13 *Anwendungsfalldiagramm für das Bestellwesen*

In dem Anwendungsfall »Waren liefern« werden alle in einer Lieferung enthaltenen Einzelposten als »geliefert« markiert und ein Lieferschein erstellt. In dem Anwendungsfall »Bezahlung entgegennehmen« wird u.a. das Zahlungsdatum der Bestellung gesetzt. Dieser Anwendungsfall wird von dem Anwendungsfall »Kreditkarte belasten« erweitert.

Der Anwendungsfall »Kunden löschen« wird automatisch vom System ausgeführt. Er löscht die Daten der zuvor vom Akteur Sachbearbeiter deaktivierten Kunden endgültig, wenn die Aufbewahrungsfrist für die Dokumente abgelaufen ist. Da nichts darüber ausgesagt ist, wie bzw. wann genau diese Funktionalität zu aktivieren ist, steht der

Anwendungsfall zu keinem Akteur in Beziehung. Wäre jedoch angegeben, dass Daten nicht mehr aktiver Kunden periodisch z. B. zu jedem letzten Monatstag zu löschen sind, würde man den Anwendungsfall »Kunden löschen« mit dem maschinellen Akteur Systemuhr (vgl. Abschnitt 12.1) verbinden.

Festzuhalten ist: Anwendungsfälle beschreiben Aufgaben, die von einem oder mehreren Akteuren mit Hilfe des Anwendungssystems durchgeführt werden. Jeder Anwendungsfall besteht aus einem »normalen« Ablauf, der um Alternativen zur Behandlung möglicher Sonderfälle und Ausnahmen ergänzt werden kann. Anwendungsfälle können über die include- oder extend-Beziehungen miteinander verknüpft und – ebenso wie Akteure – generalisiert werden. Ein Szenario ist ein bestimmter Ablauf, der bei einer konkreten Bearbeitung der Aufgabe mit dem Anwendungssystem beobachtet werden kann.

Anwendungsfälle bilden ein besonders wichtiges Modellierungskonzept in der Anforderungsermittlung. Wegen ihrer Einfachheit und Informalität stellen sie die zentrale Kommunikationsbasis zwischen Anwendern (Benutzer, Auftraggeber) und Analytikern dar. Dabei ist insbesondere auf einfache und natürliche Formulierungen zu achten. Dies ist umso wichtiger, als andere Elemente der Anforderungsspezifikation wie z. B. das Klassenmodell oder Zustandsdiagramme erfahrungsgemäß von Anwendern nur selten verstanden werden. Eine Validierung der Anforderungen durch die Anwender ist daher fast nur über Anwendungsfälle möglich, so dass ihnen auch in dieser Hinsicht eine wesentliche Rolle zukommt.