T U M

INSTITUT FÜR INFORMATIK

Internet -Buchhandel
Eine Fallstudie für die Anwendung von Softwareentwicklungstechniken mit der UML

Gerhard Popp, Franz Huber, Ingolf Krüger,
Bernhard Rumpe, Wolfgang Schwerin

TUM-I9915
September 99

TECHNISCHE UNIVERSITÄT MÜNCHEN

TUM-INFO-09-I9915-7/1.-FI

Druck: Institut für Informatik der
Technischen Universität München

INTERNET-BUCHHANDEL

EINE FALLSTUDIE FÜR DIE ANWENDUNG VON SOFT-WAREENTWICKLUNGS-TECHNIKEN MIT DER UML

Gerhard Popp
Franz Huber
Ingolf Krüger
Bernhard Rumpe
Wolfgang Schwerin

Technische Universität München

Zusammenfassung

Dieser technische Bericht beschreibt die Analyse und den Entwurf wesentlicher Kernziele eines Internet–Buchhandels. Die Entwicklung wurde unter Benutzung der UML–Notation dokumentiert und die Ergebnisse insbesondere in bezug auf die gewählte Vorgehensweise analysiert.

Bei der Entwicklung wurden spezielle Software–Entwicklungsprinzipien, wie Kapselung, Verfeinerung und Abstraktion, Unterspezifikation, Hierarchiebildung oder Generalisierung, in bezug auf Beschreibungstechniken, wie Klassendiagramme, Sequenzdiagramme und Statecharts, konsequent verfolgt und in bezug auf ihre Nützlichkeit überprüft.

Ausgehend von einem Pflichtenheft wird ein nutzungsfallorientierter Entwurf durchgeführt, der bis zu einer Verhaltensbeschreibung einzelner Objekte und einer detaillierten Klärung der Klassenstruktur führt.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Die Entwicklung von Softwareprodukten besitzt in der heutigen, stark auf Informationstechnik ausgerichteten Industrie eine außerordentliche Bedeutung. Die Erstellung industrieller Softwareprodukte ist daher ähnlich systematischen und strukturierten Richtlinien zu unterwerfen, wie es bei anderen industriellen Produkten, z.B. Autos oder Videokameras, üblich ist. Nur dadurch lassen sich Entwurfs– und Produktionsprozeß und damit die Produktqualität verbessern. In den klassischen Ingenieursdisziplinen werden deshalb ausgereifte, genormte Techniken eingesetzt, die die verschiedenen Phasen der Erstellung eines Produkts, von der Entwicklung bis zur Wartung, unterstützen. Solche Techniken bieten Produktsichten, in denen einzelne Aspekte explizit herausgearbeitet werden können. Beispiele hierzu sind etwa Schalt– oder Baupläne. Diese Techniken werden zumeist durch Werkzeuge unterstützt, die Routinearbeiten übernehmen sowie überprüfbare Entwicklungsfehler verhindern.

Weil die inhärente Komplexität von Softwaresystemen von anderen Produkten kaum erreicht wird, konnte ein einheitlicher systematischer und strukturierter Konstruktionsprozeß (in der Informatik auch Entwicklungsprozeß genannt) noch nicht gefunden werden. In der Softwaretechnik werden heute verstärkt objektorientierte Methoden eingesetzt und die Unified Modeling Language (UML, siehe z.B. [BBH+99], [BRJ98], [RJB98] [FS98], [JBR98]) wird hierfür einen neuen Standard setzen. Die UML bietet als Kombination mehrerer älterer objektorientierter Methoden eine reichhaltige Menge von Konzepten, die unter anderem in Klassen–, Zustands–, oder Sequenzdiagrammen zusammengefaßt sind und verschiedene Sichten und Abstraktionen eines Systems darstellen. Für die UML existiert derzeit eine präzise Festlegung der Syntax und eine informelle, teilweise noch unpräzise und mißverständliche Semantik. Techniken zur Transformation von UML Modellen zum Zweck der Verfeinerung oder Komposition werden derzeit von einigen Gruppen diskutiert (z.B. im SysLab–Projekt an der Technischen Universität München, der Precise UML–Group (siehe [FELR98], [EFLR99], [EBF+98], [BHP+98])), sind aber in der Praxis noch nicht etabliert. Genau diese Techniken aber bilden das Bindeglied zwischen den Methoden im Großen und den Beschreibungstechniken. Dadurch wird etwa der Projektfortschritt meßbar und besser beherrschbar. Auch können Anforderungen und Entwurfsentscheidungen besser bis zur Implementierung verfolgt werden (Traceability).

In diesem Bericht wurden diese Softwareentwicklungstechniken an einer Fallstudie angewandt und die Ergebnisse analysiert. Als umfangreiches Beispiel wurde zur Modellierung ein Internet–basierter Buchhandel gewählt, wobei es sich um eine E–Commerce–Lösung handelt, denn Angebot und Bestellungsabwicklungen können mit Hilfe des World Wide Web realisiert werden. Da ein Softwaresystem dieser Bauart mehrere komplexe Aufgabenbereiche besitzt, wurden auf verschiedenen Abstraktionsstufen die Entwürfe und die daraus resultierenden Ergebnisse mit Hilfe der bereits erwähnten Softwareentwicklungstechniken ausgearbeitet und beschrieben.

Die Entwicklungsschritte werden von einem heute in der objektorientierten Modellierung üblichen nutzungsfallorientierten Entwurfsprozeß geleitet. Diese Form des Entwurfsprozesses, der auch in [JBR98] und [Kru99] beschrieben ist, erlaubt eine stärkere Durchgängigkeit der Modellierung der Anforderungen bis zur Implementierung. Nutzungsfälle bilden dabei die gemeinsame Klammer der verschiedenen Phasen.

Die Erstellung eines vollständigen Entwurfs zum Softwareprodukt für den Internet–Buchhandel ist eine Aufgabe, die mehrere Personenjahre erfordert. Bei der Realisierung von großen Softwareprodukten arbeiten heute aus diesem Grund viele Entwickler zusammen, so daß ein anwendbarer Entwurf und damit das zu erstellende Softwareprodukt in einer angemessenen Zeit fertiggestellt wird. Da der hier erstellte Entwurf jedoch den Charakter einer Fallstudie besitzt, werden nicht alle Entwurfsschritte vollständig ausgearbeitet. Stattdessen werden in den einzelnen Schritten in erster Linie die für die folgenden Entwicklungsstufen nötigen Dokumente vollständig ausgearbeitet. Dennoch ist die Fallstudie so angelegt, daß wesentliche Teile der Entwicklung bearbeitet wurden und so eine detaillierte Analyse der Ergebnisse möglich ist.

Aus der beschriebenen Vorgangsweise bei der Erstellung der Fallstudie zum Internet–Bookshop ergibt sich die folgende Gliederung dieses Berichts. Während in diesem Kapitel das Ziel und damit sowohl die Anwendung von Softwareentwicklungstechniken wie auch des elektronischen Buchhandels im Internet mit dem Vorgangsprozeß angeschnitten wurden, wird in Kapitel 2 die detaillierte Aufgabenstellung vorgestellt. Zur UML, zu den Softwareentwicklungsprinzipien von SysLab und zum Buchhandel wird explizit eine Einführung gegeben. Der Entwicklungsprozeß und ein Zeitplan für den Arbeitsablauf werden außerdem in einem allgemeinen Abschnitt vorgestellt.

Da für einen elektronischen Internet–Buchhandel keine Beschreibung der Funktionalität, Daten und Leistungen vorlag, wurde als Ausgangspunkt für einen objektorientierten Entwurf ein klassisches Pflichtenheft erstellt. In Kapitel 3 wird eine vereinfachte Form eines Pflichtenheftes, manchmal auch Lastenheft oder grobes Pflichtenheft genannt, für die weiteren Bearbeitungsschritte erstellt. Kaptitel 4 beschäftigt sich mit der Struktur des Anwendungskerns. Dabei werden zuerst partielle Fachklassendiagramme entwickelt und im Abschluß findet eine Vereinigung dieser Teildiagramme statt.

Nutzungsfälle auf verschiedenen Abstraktionsstufen stehen in den nächsten beiden Kapiteln im Vordergrund der Betrachtung. Während sich Kapitel 5 mit der Identifikation und einer Beschreibung der Nutzungsfälle auf Systemebene beschäftigt, werden einzelne Nutzungsfälle im Kapitel 6 unter interaktionsorientierten Gesichtspunkten genauer analysiert.

Auf das Verhalten einzelner Objekte wird im 7. Kapitel näher eingegangen. Zustände und deren dynamische Veränderungen werden dort genauer betrachtet. Nachdem die Dynamik des Systems genauer spezifiziert wurde, beschäftigt sich Kapitel 8 nochmals mit der statischen Klassenstruktur. Ergänzungen, die sich bei der Analyse des dynamischen Verhaltens ergaben, werden in das Fachklassendiagramm eingebracht. Kapitel 9 enthält neben einem Ausblick die Ergebnisse der Arbeit und eine Gesamtbewertung.

Kapitel 2

Notation, Prinzipien, Vorgehensweise und Fallbeispiel

In diesem Kapitel gehen wir auf die im folgenden verwendete Notation, die Unified Modeling Language (UML), ein. Wir stellen die Softwareentwicklungsprinzipien dar, die wir im Rahmen dieser Fallstudie anwenden und erläutern unsere Vorgehensweise. Eine Einführung zu unserem Fallbeispiel schließt das Kapitel ab.

2.1 Kurzeinführung in die UML

Bei der Unified Modeling Language (UML) handelt es sich um eine Beschreibungssprache zur Spezifikation, Darstellung, Konstruktion und Dokumentation von ganzen Softwaresystemen. Die UML bietet eine Sammlung von Notationen, die sich bei der Modellierung von großen und komplexen Systemen als erfolgreich erwiesen haben (siehe [Rat97]).

Die UML ermöglicht nach [BRJ98] eine standardisierte Beschreibung von Systementwürfen auf konzeptionellen und detaillierten Ebenen. Neben der konzeptionellen Beschreibung z.B. von Geschäftsprozessen und Systemfunktionen können auch detaillierte, technische Beschreibungen angefertigt werden. Beispiele hierzu sind Klassenbeschreibungen in einer geeigneten Programmiersprache, Datenbank–Schemata und wiederverwendbare Softwarekomponenten.

Die UML ist der Nachfolger einer ganzen Anzahl von objektorientierten Analyse– und Designmethoden, die in den späten 80er und frühen 90er Jahren entstanden sind. Insbesondere vereinigt und erweitert sie die Notationen von Grady Booch (siehe [Boo94]), James Rumbaugh (siehe [RBP+91]) und Ivar Jacobson (siehe [Jac93]). Nach der ersten Phase der Diversifikation folgte eine Phase der Vereinheitlichung der objektorientierten Methoden, der auch die OMG[1] forciert wurde. Die UML 1.3 ist seit Oktober 1998 verfügbar und stellt derzeit die aktuelle Version dar. Der Standardisierungsprozeß wird demnächst mit der Version 1.4 abgeschlossen.

Die UML wird als Modellierungssprache bezeichnet. Methoden bestehen, zumindest prinzipiell, sowohl aus einer Modellierungssprache, als auch aus einer Vorgehensweise. Auf die Vorgehensweise für diese Fallstudie gehen wir in Abschnitt 2.3 auf der nächsten Seite ein.

Eine Einführung zu UML kann in folgenden Literaturstellen nachgeschlagen werden: [FS98], [Bre98a], [Bre98b], [Oes97], [Bur97] und [Neu98]. Weiterführende Literatur hierzu bieten [BRJ98], [Rat97], [BRS97], [Alh98] und [BBH+99].

2.2 Einführung zu den Softwareentwicklungsprinzipien von SysLab

Für die UML existiert derzeit eine präzise Festlegung der Syntax und eine informelle, an vielen Stellen noch unpräzise und mißverständliche, Semantik. In einem ersten Schritt ist daher eine präzisere semantische Fundierung von UML notwendig (vergleiche [EFLR98], [BGH+97b], [KR97], [KRS97], [BGH+98]). Basierend auf dem so gewonnenen Verständnis für die UML wurden bereits lange bekannte Softwareentwicklungsprinzipien auf die UML übertragen bzw. speziell auf die UML zugeschnittene Techniken für die UML neu entwickelt und mit den wenigen, bereits in der UML existierenden Techniken integriert.

Diese Softwareentwicklungstechniken stützen sich auf wesentliche Prinzipien der Softwareentwicklung ab. Die Entwicklungen haben das Ziel, Software bezüglich Korrektheit, Erhaltbarkeit, Abhängigkeit, Verwendbarkeit und Effizienz zu verbessern. Das Produkt soll eine hohe Qualität zu einem vertretbaren Preis aufweisen und auch zum vereinbarten Zeitpunkt fertig gestellt sein.

Wichtige Prinzipien sind z.B. Abstraktion, Kapselung, Komposition und Dekomposition, hierarchische Strukturierung sowie Transformationen und Verfeinerungen. Eine ausführliche Beschreibung der Prinzipien und allgemein der Softwareentwicklungstechniken ist in [BBH+99] zu finden.

Diese Prinzipien sind bei der objektorientierten Softwareentwicklung in der Bildung der Klassen und Klassenstruktur bereits relativ bekannt und manifestieren sich teilweise in der Notation selbst. Beispiele hierfür sind Generalisierung von Klassen oder private (gekapselte) Attribute.

Jedoch können diese Prinzipien auch auf andere Beschreibungstechniken angewendet werden, wie zum Beispiel die Hierarchiebildung oder die Generalisierung–/Verfeinerungstransformation auf den Statecharts (vgl. [Rum96]) und auf den Sequenzdiagrammen (siehe

[1] Object Management Group, www.omg.org

[Krü99]) sowie Techniken zum Übergang von Sequenzdiagrammen zu Statecharts (vgl. [KGSB99]) zeigen.

2.3 Die Entwicklungsmethode

In dieser Fallstudie wird die in [Bre98a] beschriebene Methode MOS[2] verwendet. Der dort vorgeschlagene Prozeß verwendet die UML als Modellierungs– bzw. Beschreibungssprache. Im Rahmen der Vorgehensweise der Methode wird beschrieben, welche Aktivitäten durchzuführen sind, welche Ergebnisse sie liefern und wie sie zusammenhängen. Dies umfaßt den Einsatz von Notationen zur Erstellung von Sichten, die Übergänge zwischen Sichten und die systematische Anwendung von Prinzipien. Die Methode, und damit der Prozeß, hier die MOS–Methodik, besitzt teilweise Analogien zu den in [FS98], [Jac93], [CAB$^+$94] und [JBR98] vorgestellten Methoden.

Im Vordergrund steht hier eine nutzungsfallorientierte Vorgehensweise. Diese wurde in [Jac93] vorgestellt und wird heute sehr häufig in Zusammenhang mit objektorientierter Softwareentwicklung eingesetzt. Booch, Jacobson und Rumbaugh verwenden in ihrem neuen Buch „The Unified Software Development Process“ (siehe [JBR98]) ebenfalls diese Entwurfsmethode. Nutzungsfälle (engl. „Use Cases“) werden auch in [FS98], [CAB$^+$94] und [BK$^+$96] verwendet.

Der nutzungsfallorientierte Entwicklungsvorgang dieser Fallstudie stammt in erster Linie aus [Bre98a]. Ausgehend von einem klassischen Pflichtenheft (siehe [Bal96]) werden zu den Produktdaten ein Fachklassendiagramm und zu den Produktfunktionen Nutzungsfälle entwickelt. Die Nutzungsfälle werden zuerst auf der Ebene des Systems ermittelt. Auf dieser Ebene wird das System als „black box“ betrachtet und Folgen von Nutzungsfällen beschrieben. Anschließend daran werden zusätzlich zu den Fachklassen sogenannte Vorgangsklassen eingeführt, die den Ablauf eines Nutzungsfalls steuern. Darauf aufbauend kann für diesen einzelnen Nutzungsfall beschrieben werden, welche Interaktionen zwischen Nutzer, Vorgangs– und Fachklassen ablaufen.

Dieser Ablauf spiegelt sich auch in dem in Abbildung 2.1 auf der vorherigen Seite dargestellten Zeitplan wider.

Hinweis zu kursiv gedruckten Texten: An einigen wenigen Stellen wurden Anmerkungen z.B. zu einem Diagramm ergänzt oder Schwierigkeiten beim Entwurf erläutert. Solche Textpassagen wurden im Diagramm zur Hervorhebung kursiv dargestellt.

[2] Methode – Objekte – Spezifikation

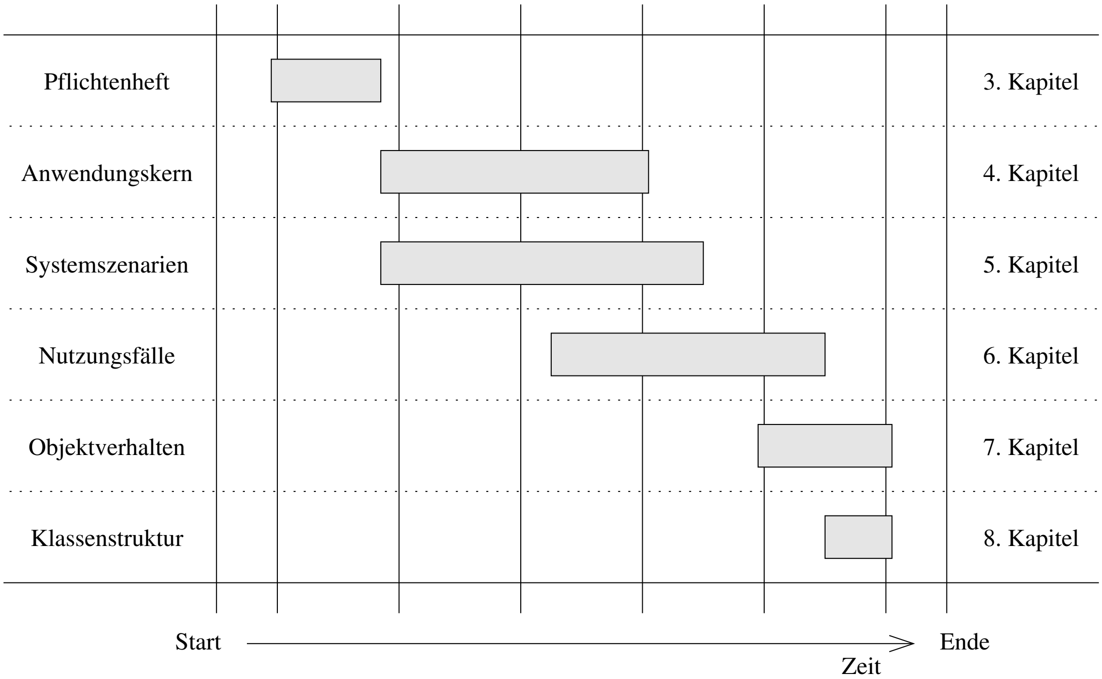


Abbildung 2.1: Zeitplan der Fallstudie

2.4 Das Beispiel: Internet–Buchhandel

Beim Internet–Buchhandel[3] bietet ein Buchhandel seine Dienstleistung über ein Netzwerk mit geeigneten Endeinrichtungen (z.B. Personal Computer mit Betriebssystem und Web–Browser) an. Der Kunde kann sich kostenlos über verfügbare Bücher informieren und gegebenenfalls bestellen. Zum Bestellen werden die Bücher in einem virtuellen Einkaufskorb abgelegt und bei der Bestellaufgabe werden die zur Lieferung und Bezahlung notwendigen Daten eingegeben. Neben einer Suche als Büchersuche bzw. Blättern im Katalog kann ein Anwender zu einem gelesenen Buch noch eine Kritik oder Empfehlung abgeben.

Da die Bücher über das Netz bestellt werden können, die Auslieferung und Bezahlung aber auf herkömmliche Weise stattfinden, handelt es sich hierbei um eine E–Commerce–Lösung. Nach [CRF98] handelt es sich um E–Commerce dann, wenn eine Transaktion über das Netz initiiert, aber nicht vollständig im Web abgewickelt wird.

E–Commerce–Anwendungen, wie z.B. der Internet–Bookshop, finden durch die Verbreitung von Rechnung mit Internetzugang sowohl im Heimbereich wie auch in Firmen wachsende Anwendung. Der Anwender kann bequem von zu Hause, schnell und unkompliziert ohne Einhaltung von irgendwelchen Öffnungszeiten Dienstleistungen nutzen oder Warenbestellungen aktivieren.

[3] In diesem Bericht wird an Stelle des deutschen Begriffs Buchhandel auch der in der englischen Sprache entsprechende Begriff „Bookshop“ verwendet.

Da tatkräftig an der Einführung von Internet–Währungen gearbeitet wird, soll dies im Entwurf an geeigneten Stellen berücksichtigt werden. Solange Internet–Währungen wie NetCash, DigiCash oder CyberCoin noch nicht verwendet werden können, kann z.B. mit einer Kreditkarte bezahlt werden. Auch hierfür müssen sichere Übertragungen z.B. der Kartennummer gewährleistet sein. Die Kreditkartenanwendung im Internet wird in [Lan97] und [CRF98] beschrieben. Internetwährungen sind in [DD97], [DD98], [Dre98] und [Lan98] zu finden. Zum Bookshop können weiterführende Informationen in [Mic99], zum weltweiten Internet–Handel in [Bey98] nachgeschlagen werden.

Kapitel 3

Pflichtenheft

Dieses Kapitel stellt alle fachlichen Basisanforderungen, die das zu entwickelnde Software–Produkt aus der Sicht des Auftraggebers erfüllen muß, in einem sogenannten Pflichtenheft zusammen. Die hier verwendete Form des Pflichtenheftes wird in [Bal96] als „grobes Pflichtenheft“ oder als „Lastenheft“ bezeichnet. Im Gegensatz zu dem in [Bal96] oder [PB96] bezeichneten „Pflichtenheft“ enthält das hier verwendete Pflichtenheft eine ungenauere Spezifikation der Abgrenzungskriterien, des Produkt–Einsatzes und –Umfangs und der Produktfunktionen, da sich diese Spezifikationen erst im Verlauf des objektorientierten Entwurfs exakt klären lassen.

Die jeweils kursiv gedruckten Informationen unter den folgenden Gliederungspunkten dienen nur zur Aufgabenklärung. Sie sollen erläutern, was unter dem jeweiligen Punkt anzuführen ist.

Das Pflichtenheft dient in den weiteren Kapiteln in erster Linie als Grundspezifikation für die Daten und Funktionen des Systems. Die Produktdaten sind Grundlage zur Erstellung eines Fachklassendiagramms und aus den Produktfunktionen entstehen die Nutzungsfälle. Die Produktleistungen, Qualitätsanforderungen und Ergänzungen sowie die Zielbestimmung und der Produkteinsatz dienen zur Klärung der Produktanwendung und weiterer Produkteigenschaften. Die folgenden Kapitel umfassen in erster Linie die Modellierung von Produktfunktionen und Produktdaten. Die methodische Umsetzung von nicht funktionalen Anforderungen, die auch im Pflichtenheft enthalten sind, steht dabei nicht im Vordergrund.

3.1 Zielbestimmung

An dieser Stelle wird beschrieben, welche Ziele durch den Einsatz des Produktes erreicht werden sollen.

Durch das zu erstellende Produkt soll es ermöglicht werden, Bücher über das Internet zu verkaufen. Diese werden dann durch einen oder mehrere Paketdienste ausgeliefert und mittels Kreditkarte, Bankeinzug oder Scheck bezahlt. Hierbei handelt es sich um eine E–Commerce–Lösung, denn es werden Waren mit Hilfe des Internets gehandelt. E–Commerce verlangt nicht, daß alle Transaktionen über das Netz abgewickelt werden.

Künftige allgemein akzeptierte Internetwährungen wie z.B. NetCash, DigiCash oder CyberCoin könnten den elektronischen Handel noch weiter vervollständigen. Die Architektur des Systems ist bezüglich solcher Entwicklungen flexibel zu halten.

3.2 Produkteinsatz

Legt fest, für welche Anwendungsbereiche und für welche Zielgruppen das Produkt vorgesehen ist.

Der Internet–Bookshop dient in erster Linie dem Verkauf von Büchern. Eine Büchersuche zu einem bestimmten Interessengebiet sowie die gezielte Suche nach einem Buch soll neben einer Bestellung und der Festlegung der Zahlungsabwicklung genauso vorhanden sein wie eine Benutzerinformation über die Verfügbarkeit der Lieferung. Genaue Preisangaben zum Produkt, hier ein Buch, und Informationen über eventuelle Lieferzuschläge sollen dem Benutzer ebenfalls zugänglich gemacht werden. Weiterhin ist es Ziel des zukünftigen Produkts, alle Aufgaben des Händlers wie Bestellung beim Großhändler, Rechnungsabwicklung, Versandabwicklung, Bankanbindung und Verbindung mit dem Steuerberater zu unterstützen.

Zielgruppe des Produkts sind Internet–Benutzer im In– und Ausland. Es können sowohl Privatleute als auch Firmen vom Internet–Bookshop beliefert werden. Das Händler–Personal arbeitet ebenfalls mit dem zu erstellenden Produkt.

3.3 Produktfunktionen

Die Hauptfunktionen des Produktes werden aus Auftraggebersicht beschrieben. Es ist darauf zu achten, daß die Kernfunktionen und nicht sekundäre Funktionen beschrieben werden. Auf Detailbeschreibungen ist zu verzichten. Jede Funktionsanforderung ist durch einen eindeutigen Bezeichner mit vorangestelltem F zu markieren, um eindeutig referenziert werden zu können.

- Kundensicht

  /F:Suche/ Büchersuche, ausgehend von Autor, Titel, ISBN-Nummer oder Interessengebiet. Hierbei Unterscheidung zwischen Laien– und Expertensuche.

/F:Blättern/ Blättern im Bücherkatalog. Alle lieferbaren Artikel bzw. eine Selektion nach obigen Kriterien (/F:Suche/) können sequentiell durchgeblättert werden. Hierzu sind verschiedene Sortierreihenfolgen nach Autor, Titel und Interessen vorteilhaft.

/F:Vormerken/ Vormerken zum Kauf. Ein gefundenes Buch kann in einem „Einkaufskorb“ abgelegt werden. Dadurch wird es ermöglicht, mehrere Bücher gemeinsam zu kaufen und eventuell Bücher wieder abzuwählen, wenn z.B. ein geeigneteres gefunden wird oder wenn der Gesamtpreis des Einkaufs zu hoch geworden ist. Der Einkaufskorb kann jederzeit in Form einer Liste betrachtet werden.

/F:Bestellung/ Bestellung der im Einkaufskorb liegenden Bücher. Die vorher ausgewählten Bücher werden jetzt definitiv bestellt. Der Käufer muß dem Dienstleister hierzu Informationen über seine Person, über die Zahlungsweise und über die Modalitäten der Lieferung bekanntgeben. Bezahlt werden kann mittels Kreditkarte, Scheckeinreichung, Lastschrifteinzug oder in Zukunft mit einer akzeptierten Internet–Währung. Personen mit guter Zahlungsmoral können besondere Preisbedingungen eingeräumt werden.

/F:Lesermeinung/ Schreiben von Lesermeinungen. Die Käufer können zu ihren Büchern Empfehlungen und Kritiken abgeben, die dann im Katalog gespeichert und auf den jeweiligen Buchseiten angezeigt werden.

- Händlersicht

/F:Kundenverwaltung/ Verwaltung der Kunden. Die Adresse des Kunden inklusive Email für die Zusendung der Bücher sowie die Bankverbindung bzw. die Kreditkartennummer werden gespeichert. Für spätere Einkäufe wird auch ein Paßwort vergeben, damit bei nachfolgenden Bestellungen nicht nochmals Angaben zur Person abgefragt werden müssen. Als Benutzerkennzeichen dient die Email–Adresse des Kunden, da sie im allgemeinen eindeutig ist. Eventuelle Änderungen müssen jedoch möglich sein. Die Summe der Gesamteinkäufe des Kunden und die Zahlungsmoral werden vermerkt. Dadurch kann der Kunde später bevorzugt werden (Preisnachlaß) oder bei wiederholt verspäteter Zahlung von besonderen Bezahlungsbedingungen ausgeschlossen werden. Kunden, die über Jahre nichts mehr gekauft haben, werden aus der Kundenkartei gelöscht.

/F:Bestellungsverwaltung/ Interne Bestellungsverwaltung zum Internet–Bookshop. Die eingegangenen Bestellungen müssen auf Verfügbarkeit überprüft werden. Eventuell müssen vom Großhändler oder Verlag Bücher angefordert werden. Sobald alle Bücher eines Einkaufs zur Verfügung stehen, wird ein Auftrag an die Auslieferungsabteilung übermittelt. Bei zu langen Wartezeiten und großen Bestellungen ist auch eine Teillieferung möglich.

/F:Stornierung/ Stornierung der Bestellungen oder von Teilbestellungen. Sollte der Kunde Irrtümer in seiner Bestellung erkennen, so kann er diese dem Internet–

Bookshop mittels Email, Fax oder Telefon mitteilen. Die Stornierungseingabe findet aber nur durch Mitarbeiter des Bookshops statt und kann somit nicht direkt vom Käufer eingegeben werden. Dies soll verhindern, daß sich eine wiederholte Bestellung und Abbestellung zum Regelfall entwickelt.

/F:Statistik/ Über die verkauften Bücher soll eine Statistik erstellt werden. So können z.B. Bücher mit hohen Verkaufszahlen oder hohem Vorrat auf den Einstiegsseiten zu Werbezwecken abgebildet werden. Außerdem können dann von den häufig verkauften Büchern immer größere Bestände beim Großhändler bestellt und zwischengelagert werden.

/F:Katalogbearbeitung/ Die Bearbeitung des Bücherkataloges. Die Eingabe von Neuerscheinungen, Preisänderungen, Verfügbarkeitsänderungen und das Löschen von Büchern im Auswahlkatalog wird durch das Personal erledigt.

/F:Rechnungserstellung/ Erstellen von Rechnungen. Das neue Produkt soll auch die Rechnungen zu den Lieferungen verwalten und drucken.

/F:Etikettendruck/ Etiketten, die auf die zu versendenden Päckchen aufgeklebt werden, sollen ebenfalls beim Abwickeln eines Auftrags mitgedruckt werden.

/F:Datenaustausch/ Austausch von Daten mit dem Großhändler, dem Steuerberater, den Banken und der Lagerverwaltung. Bestellungsdaten und Katalogdaten sollen z.B. über das Internet zum und vom Großhändler übermittelt werden können, so daß diese nicht erneut an einem Rechner eingegeben werden müssen. Eine Schnittstelle zum Steuerberater ermöglicht eine einfache und korrekte Steuerabrechnung. Die für die Auslieferung relevanten Daten werden an die Lagerverwaltung weitergegeben.

/F:Großhändlerverwaltung/ Verwaltung der Großhändler. Daten von Großhändlern, von denen Bücher bezogen werden, müssen hinzugefügt, gelöscht und geändert werden können.

/F:Paketdienstverwaltung/ Verwaltung der Paketdienste. Eine Liste der verfügbaren Auslieferungsdienste muß geändert, ergänzt und reduziert werden können.

3.4 Produktdaten

Hauptdaten des Produktes, die permanent gespeichert werden müssen, werden festgelegt. In Analogie zu den Produktfunktionen werden sie mit einem D gekennzeichnet.

/D:Kunden/ Kundendaten. Anschrift, Email und Bankverbindung der Kunden sowie die Zahl der bisher geleisteten Einkäufe und die Gesamteinkaufssumme müssen gespeichert werden. Von besonderer Bedeutung erweist sich auch die Zahlungsmoral des Kunden. Für eine wiederholte Bestellung muß das vergebene Paßwort verwaltet werden. Ein eigenes Login ist nicht nötig, da hierzu die Email–Adresse verwendet wird.

/D:Firmenkunden/ Bei Firmen sind die Firmendaten und die Daten der Kontaktperson zu speichern. Besondere Preis– und Lieferbedingungen sind hier ebenfalls zu vermerken.

/D:Einkaufskorb/ Einkaufskorb. Die Liste der ausgewählten und noch nicht bestellten Bücher wird im Einkaufskorb verwaltet. Sobald eine Sitzung ohne abschließende Bestellung beendet wird, werden die Einträge gelöscht.

/D:Bestellungen/ Bestelldaten. Die aktuellen Aufträge werden für die Bestellabwicklung gespeichert. Auch alte Aufträge (z.B. der letzten fünf Jahre) werden ebenfalls im System gehalten, damit eine aussagekräftige Statistik geführt und Angaben zum Kaufverhalten des Kunden berechnet werden können.

/D:Buchkatalog/ Der Buchkatalog mit dem Lagerbestand oder den Vertriebspartnern ist zu speichern. Außerdem werden hier die Statistikergebnisse und die Empfehlungen/Kritiken abgelegt.

/D:Großhändler/ Eine Liste der Großhändler, von denen die Bücher bezogen werden, muß ebenfalls verwaltet werden.

/D:Paketdienste/ Die Paketdienste mit ihren Lieferbedingungen, -kosten und Auslieferungszeiten werden gespeichert.

3.5 Produktleistungen

Werden an einzelne Hauptfunktionen und Hauptdaten Leistungsanforderungen bzgl. Zeit, Datenumfang oder Genauigkeit gestellt, dann werden sie hier aufgeführt. Den Referenzbezeichnern wird ein L vorangestellt.

/L:Einträge im Buchkatalog/ Im Buchkatalog werden ca. 1.000.000 Einträge abgelegt. (Beim elektronischen Buchversand „Amazon.de“[1] gibt es zur Zeit etwa 335.000 Bücher.) Hieraus ergibt sich die Forderung für einen großen Massenspeicher mit schneller Zugriffszeit.

/L:Schnelle Katalogsuche/ Auf den Katalogdaten muß eine schnelle Suche nach verschiedenen Kriterien wie z.B. ISBN–Nr., Titel und Autor durchgeführt werden können.

[1] Im World Wide Web ist der Buchversand „Amazon.de“ unter der URL „`http://www.amazon.de`“ zu finden.

/L:Sichere Speicherung/ Die Personendaten, besonders die Daten der Bankverbindung bzw. der Kreditkarte müssen genau und sicher übertragen und gespeichert werden. Auch die Anschrift muß für eine exakte Zustellung einwandfrei einer Person zugeordnet werden können und fehlerfrei sein.

3.6 Qualitätsanforderungen

Die wichtigsten Qualitätsanforderungen werden hier aufgeführt, wie gute Zuverlässigkeit, gute Benutzbarkeit, Effizienz usw.

Produktqualität	sehr gut	gut	normal	nicht relevant
Funktionalität		X		
Zuverlässigkeit		X		
Benutzbarkeit		X		
Effizienz		X		
Änderbarkeit			X	
Übertragbarkeit				X
Sicherheit	X			

Tabelle 3.1: Qualitätsanforderungen

3.7 Ergänzungen

Hier werden Ergänzungen oder spezielle Anforderungen beschrieben, z.B. außergewöhnliche Anforderungen an die Benutzerschnittstelle.

Das fertige Produkt soll mit einer sehr ansprechenden Benutzeroberfläche ausgeliefert werden. Dadurch soll Aufmerksamkeit beim Kunden geweckt werden sowie das Bedürfnis, sich eingehend mit der Dienstleistung auseinanderzusetzen und diese zu verwenden.

Kapitel 4

Die Struktur des Anwendungskerns

Ausgehend vom verbalen Pflichtenheft erfolgt in diesem Kapitel eine Modellierung von Klassen des Anwendungsbereichs mit objektorientierten Techniken. In einem ersten Schritt lassen sich Klassen des Anwendungsbereichs aus den Produktdaten ableiten. Bei der Bearbeitung findet eine Modellierung der Klassen, von Generalisierungs– und Spezialisierungsbeziehungen und von Assoziationen zwischen Klassen statt.

Die Definition des Anwendungskerns spielt nach [Bre98a] eine zentrale Rolle in der Systementwicklung. Klassen, Attribute und Assoziationen werden identifiziert und zu einem Fachklassendiagramm vereinigt. Der folgende Entwurf des Fachklassendiagramms spielt für die gesamte Anwendung eine entscheidende Rolle, denn alle weiteren Beschreibungs– und Modellierungskonzepte beziehen sich auf dieses Diagramm, sei es einerseits, um Daten zu visualisieren oder andererseits, um Daten zu manipulieren, d.h. Daten im System zu löschen, verändern oder neu hinzuzufügen. Da die gesamten Bearbeitungsschritte, die später in den Nutzungsfällen genauere Betrachtung finden, zum Sinn und Zweck der Datenmodifikation entwickelt werden, findet dadurch ein ständiges Hinzufügen, Löschen und Ändern von Fachklasseninstanzen statt.

Ausgangspunkt für die Identifizierung der Klassen und Attribute sind die Produktdaten, die im Abschnitt 3.4 des Pflichtenheftes beschrieben wurden (siehe Seite 14). Zunächst werden die Produktdaten in Form von Klassen im objektorientierten Sinn abgebildet. Diese so entstehenden Klassen werden, wenn notwendig, im weiteren Entwicklungsprozeß noch transformiert und verfeinert. Gründe für eine nochmalige Betrachtung, die zu einer Verfeinerung führen, sind z.B. das Extrahieren von wiederverwendbaren Komponenten oder die Zusammenführung von unterschiedlichen Klassen mit den selben Eigenschaften. Derartige Transformationen sind insbesondere für eine iterative Vorgehensweise typisch.

Da es wenig sinnvoll wäre, alle Daten gemeinsam in einem unübersichtlichen Diagramm zu modellieren, werden zuerst immer nur Teile der Produktdaten als Klassenstruktur modelliert. (Abschnitt 4.1 auf der nächsten Seite). Die so entstehenden partiellen Fachklassendiagramme werden dann im Abschnitt 4.3 auf Seite 49 zu einem gemeinsamen Fachklassendiagramm zusammengefügt. Abschnitt 4.2 auf Seite 34 beschäftigt sich mit einer Designentscheidung zur rekursiven Teilbestellung.

4.1 Erfassung der partiellen Fachklassendiagramme

In den folgenden Teilabschnitten werden die partiellen Fachklassendiagramme zu den Kunden (4.1.1), zum Einkaufskorb (4.1.2), zu den Bestellungen (4.1.3), zu dem Buchkatalog (4.1.4), zum Vertrieb (4.1.5) und zu den Paketdiensten (4.1.6) identifiziert. Sind die partiellen Klassendiagramme bereits etwas umfangreicher, dann werden zuerst Teile davon alleinstehend modelliert und am Ende des jeweiligen Teilabschnitts zu einem Ganzen zusammengefügt.

Schon während der Erstellung der partiellen Diagramme wird versucht, ein „Refactoring“ bereits in diesem Entwicklungsstadium anzuwenden. Es wird versucht, die Klassenstruktur zu den Produktdaten bereits bezüglich der Redundanz von Daten optimal zu gestalten.

Als Refactoring werden Entwurfsschritte bezeichnet, die eine Klassenstruktur verändern, unter Erhaltung der Information und des Verhaltens der Klassen. Dabei werden Umstrukturierungen an der Klassenstruktur vorgenommen, die beispielsweise notwendige Erweiterungen ermöglichen oder Wiederverwendung unterstützen. Refactoring–Methoden können z.B. das Definieren einer abstrakten Superklasse, die Spezialisierung einer Klasse durch Einfügen von Superklassen, das Verschieben von Klassen, Attributen oder Methoden innerhalb einer Klassenhierarchie oder die Umbenennung von Klassen, Methoden und Funktionen sein. Refactoring von Klassen wird in [Opd92] ausführlich beschrieben. Im folgenden werden vor allem die Bildung von Superklassen und das Verschieben von Attributen angewendet.

4.1.1 Privat– und Firmenkunden

Ein Fachklassendiagramm, das nur die Kundendaten modelliert, wird in diesem Abschnitt schrittweise erstellt. Kundendaten können die für einen Bestellvorgang notwendigen Daten eines Privat– oder eines Firmenkunden sein. Die hierzu nötigen Daten wurden im Pflichtenheft unter Produktdaten (Abschnitt 3.4 auf Seite 14) in den Punkten */D:Kunden/* und */D:Firmenkunden/* bereits verbal zusammengestellt.

Ziel der Modellierung von Privat– und Firmendaten soll ein Fachklassendiagramm sein, welches ein möglichst redundanzfreies Speichern von Kundendaten jeglicher Art erlaubt. Da bei einer Firma mehrere Personen mit dem Buchhandel in Verbindung treten können, muß auch eine Verwaltung von mehreren Kontaktpersonen zu einer Firma ermöglicht werden.

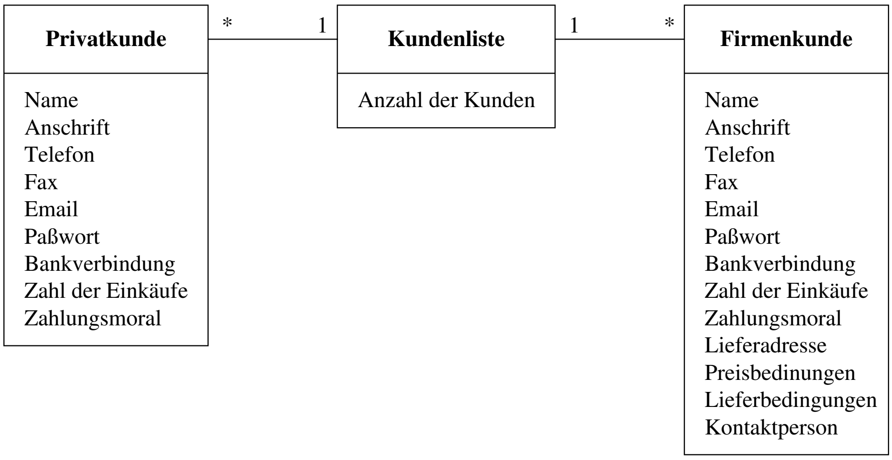


Abbildung 4.1: Fachklassendiagramm zu den Privat– und Firmenkunden

Abbildung 4.1 auf der nächsten Seite zeigt das partielle Fachklassendiagramm der Privat– und Firmenkundendaten nach der Ableitung der wesentlichen Daten aus den verbal beschriebenen Produktdaten. Die Fachklasse Kundenliste, die später z.B. Indexe für ein schnelles Auffinden von Kunden bereitstellen kann, ist mit zwei weiteren Klassen verbunden. In der assoziierten Klasse Privatkunde werden alle notwendigen persönlichen Daten für die Bestellung einer einzelnen Person abgeleitet. Die Klasse Firmenkunde enthält entsprechend die notwendigen Daten in dem Fall, daß eine Firma eine Bestellung aufgibt. Für spätere Berechnungen, Listendurchläufe und Suchanfragen ist es sinnvoll, die Anzahl der Kunden in der Klasse Kundenliste zu vermerken.

Da jedoch einige Attribute sowohl in der Klasse Privatkunde als auch in der Klasse Firmenkunde vorliegen und sich eine Zweiteilung der Kundenliste in Privat– und Firmenkunden als nicht notwendig erweist, kann die Klassenstruktur dahingehend transformiert werden.

Wird ein abstrakter Kunde eingeführt, kann die zugehörige Klasse eine Zusammenfassung von gemeinsamen Attributen und später auch Operationen der Privat– und Firmenkunden enthalten. Die speziellen Daten der Firmen– und Privatkunden werden dadurch in das System integriert, daß die abstrakte Kunden–Klasse sowohl in eine Klasse Privatkunde als auch in eine Klasse Firmenkunde abgeleitet wird. Ein Kunde ist die verallgemeinerte Art der Privat– und Firmenkunden. Privat– und Firmenkunden sind also eine spezielle Art der Kunden. Von den Subklassen Privat– und Firmenkunde zur Superklasse Kunde liegt eine IS–A–KIND–OF–Beziehung vor, d.h. es handelt sich hierbei um eine Generalisierungs–Beziehung. Opdyke bezeichnet das Zusammenfassen gemeinsamer Eigenschaften mehrerer Klassen in einer Superklasse in [Opd92] als „Refactoring to Generalize: Generating an Abstract Superclass".

Im Fachklassendiagramm werden abstrakte Klassen, wie hier die Klasse Kunde, *kursiv* dargestellt. Abstrakte Klassen können bei einer späteren Implementierung nicht instantiiert werden, d.h. es können nur Objekte von nicht abstrakten Subklassen angelegt werden.

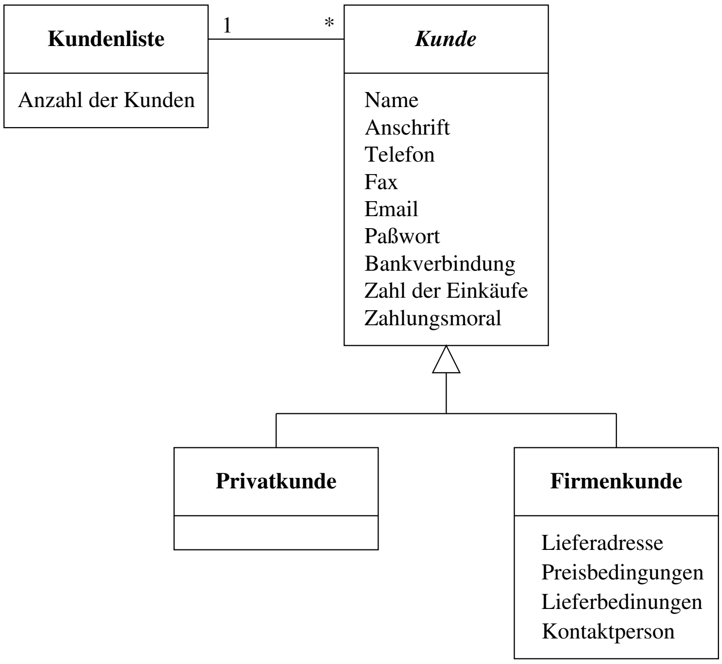


Abbildung 4.2: Privat– und Firmenkundenklassen nach der 1. Transformation

Abbildung 4.2 auf der nächsten Seite zeigt das Klassendiagramm nach der beschriebenen Transformation. Die Klasse Privatkunde enthält nun keine eigenen Attribute, da diese alle in die Superklasse Kunde gewandert sind, jedoch können hier später spezielle Operationen und Attribute zur Verfeinerung der Superklasse hinzugefügt werden (vgl. [Rum96]).

Die nun vorherrschende Struktur erlaubt zwar eine geeignete Speicherung von Firmen– und Privatkunden, aber zu jedem Firmenobjekt kann nur eine einzige Kontaktperson gespeichert werden. Hierzu enthält die Klasse Firmenkunde ein Attribut Kontaktperson. Zu beachten ist an dieser Stelle, daß noch keine Entscheidung darüber getroffen wurde, ob dem Attribut ein Werttyp oder eine Klasse als Typ zugeordnet wird. Die Attribute sind bis jetzt noch nicht typisiert. Grundsätzlich erfolgt eine Typisierung entweder dadurch, daß ein Attribut in eine Assoziation mit einer evtl. einzufügenden Klasse umgewandelt wird, oder durch Angabe eines Werttyps als Attributtyp. Werttypen sind dadurch charakterisiert, daß bei ihren Elementen nicht zwischen Identität und Zustand unterschieden wird.

Gesetzt den Fall, daß eine Firma mindestens zwei verschiedene Ansprechpartner hat, muß für jede Kontaktperson ein neuer Firmenkunde angelegt werden und somit würden die Daten wie Name, Bankverbindung, Lieferadresse (z.B. des Wareneingangs), Preis– und Lieferbedingungen redundant gespeichert. Ein Erzeugen weiterer Firmenkundenobjekte stellt hier somit nicht das gewünschte Ergebnis dar.

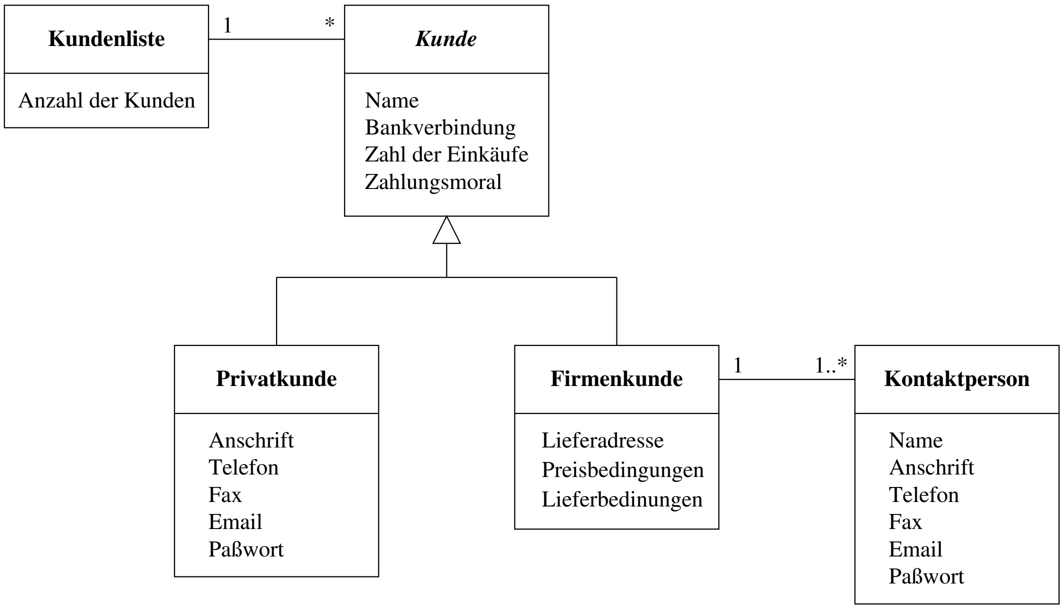


Abbildung 4.3: Privat– und Firmenkundenklassen nach der 2. Transformation

Zur Lösung dieses Problems wird das auf ein Wertobjekt verweisende Attribut Kontaktperson aus der Klasse Firmenkunde entfernt und eine eigene Klasse Kontaktperson erzeugt, deren Instanzen von einem Firmenkundenobjekt mehrfach referenziert werden können (Referenzobjekt). In einem Kontaktpersonenobjekt können nun die Attribute Name, Anschrift, Telefon, Fax, Email und das Zugangspaßwort für jeden Ansprechpartner belegt werden, und somit sind diese Attribute im Kundenobjekt für den Firmenkunden überflüssig, mit Ausnahme des Attributs Name, welches beim Firmenkunden den Firmennamen selbst enthält. Da jedoch der Privatkunde diese Angaben benötigt, und sie somit nicht entfernt werden können, findet wieder ein Verschieben der Attribute Anschrift, Telefon, Fax, Email und Paßwort zurück in die Klasse Privatkunde statt.

In der Abbildung 4.3 wird das Ergebnis der zweiten Transformation am Kunden–Fachklassendiagramm gezeigt. Die Struktur erlaubt nun ein redundanzfreies Speichern von Firmenkunden mit mehreren Kontaktpersonen sowie von Privatkunden.

Informationen über die Bankverbindung können im System gespeichert werden. Beim Bankeinzug wird der Kontoinhaber, falls dieser vom Kundennamen abweicht, der Name der Bank, die Kontonummer und die Bankleitzahl gespeichert. Bei einer Kreditkarten–Zahlungsweise wird ebenfalls, wenn notwendig, der Kontoinhaber vermerkt und es müssen die Kreditkartennummer und das Gültigkeitsdatum angegeben werden. Bei einer Scheck–Bezahlung werden keine Daten vermerkt, da sich diese stets ändern.

Die Anschrift enthält die gewöhnlichen Ortsangaben. Zusätzlich wurde noch ein Attribut Abteilung aufgenommen, in dem z.B. bei einer Firma ein Gebäudeteil oder eine Zimmer-

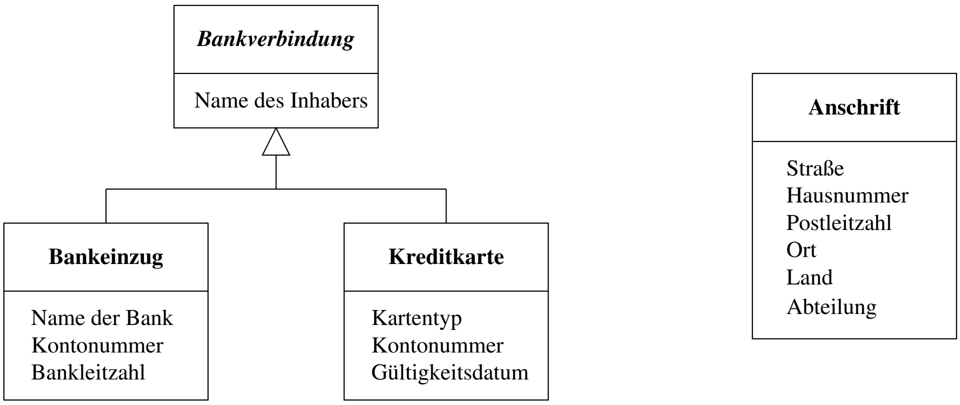


Abbildung 4.4: Klassendiagramm zur Bankverbindung und Anschrift

nummer vermerkt werden kann. Abbildung 4.4 auf der nächsten Seite zeigt das Fachklassendiagramm zur Bankverbindung und die Fachklasse Anschrift.

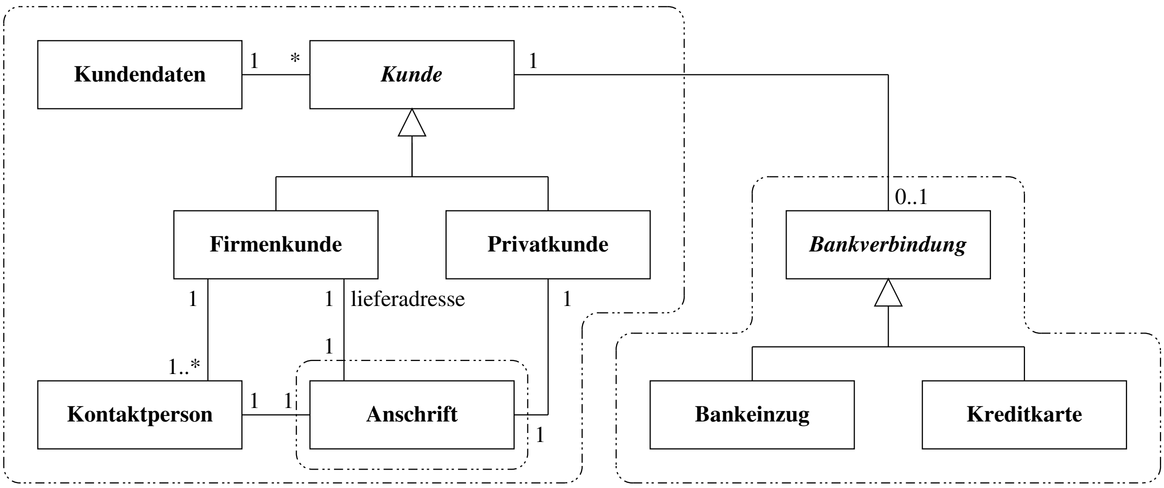


Abbildung 4.5: Erweitertes Fachklassendiagramm zu den Kunden

Zum Abschluß werden die Fachklassendiagramme aus den Abbildungen 4.3 und 4.4 in ein erweitertes Fachklassendiagramm zu den Privat– und Firmenkunden vereinigt. Die Attribute Anschrift in der Klasse Privat– und Firmenkunde, Lieferadresse in der Klasse Firmenkunde und Bankverbindung in der Klasse Kunde wurden als Referenzattribute ausgeprägt und finden sich durch entsprechende Assoziationen wieder. Abbildung 4.5 zeigt das Diagramm mit den Referenzen. Die Attribute wurden aus Übersichtsgründen weggelassen[1].

[1]Die strichpunktierten Linien kennzeichnen die vorher entwickelten Teil–Fachklassendiagramme.

4.1.2 Einkaufskorb

Wie im Abschnitt Produktdaten (siehe 3.4 auf Seite 14) des Pflichtenheftes unter dem Punkt */D:Einkaufskorb/* beschrieben wurde, enthält der Einkaufskorb eine Liste der ausgewählten und noch nicht bestellten Bücher. Ein leerer Einkaufskorb wird beim Starten einer Kunden–Sitzung erzeugt. Abbildung 4.6 zeigt das Fachklassendiagramm des Einkaufskorbs.

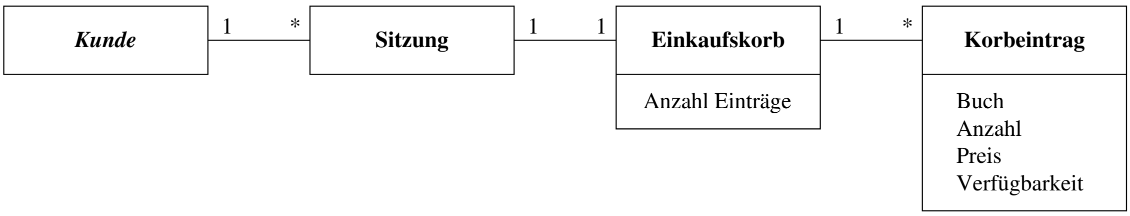


Abbildung 4.6: Fachklassendiagramm zum Einkaufskorb

Als zentrales Element enthält der Einkaufskorb eine Liste, die die bereits ausgewählten Bücher als sogenannte Korbeinträge festhält. Neben dieser Liste wird auch noch die Anzahl der Listeneinträge im Einkaufskorb verwaltet.

Da während eines Auswahlvorgangs eines Kunden auch der Buchkatalog von dem Händlerpersonal geändert werden kann, wäre auch eine Preisänderung des Buches denkbar, so daß sich der Preis bei der Bestellauswahl von dem bei der definitiven Bestellung eines Buches unterscheiden könnte. Zur Vermeidung von solchen Unstimmigkeiten muß der Kunde bei einer späteren Bestellung das Buch zu dem Preis zum Zeitpunkt der Buchauswahl bekommen. Für diesen Zweck wird der Preis im Korbeintrag zusätzlich gespeichert, obwohl der Buchpreis auch über die Buchreferenz ermittelt werden könnte. Die Verfügbarkeit kann sich ebenfalls ändern, bei einer Aufnahme in den Einkaufskorb wird jedoch die nötige Anzahl für den Kunden reserviert. Falls das Buch in der gewünschten Zahl im Lager vorrätig ist, wird das Verfügbarkeitsattribut gesetzt.

Die Einkaufskorb–Klasse besitzt neben der Korbeintrag–Assoziation auch eine Assoziation zu einer Sitzung. Die Sitzungsklasse verwaltet die Klasse Einkaufskorb, denn beim Beginn einer Kunden–Sitzung muß ein leerer Einkaufskorb erzeugt werden. Beendet ein Kunde eine Sitzung ohne definitiv zu bestellen, werden die Einträge im Einkaufskorb beim Zerstören des Einkaufskorbobjekts zusammen mit dem Sitzungsobjekt gelöscht. Nach [Bre98a] handelt es sich um eine IST–TEIL–VON–Beziehung.

In [FS98] wird eine IST–TEIL–VON–Beziehung als Komposition bezeichnet, wenn die Lebenszeit des Einkaufskorbs vom Sitzungsobjekt abhängt und ein Einkaufskorbobjekt genau einem Sitzungsobjekt zugeordnet werden kann. Es wäre auch denkbar, daß das Einkaufskorbobjekt nur über das Sitzungsobjekt angesprochen wird. Abbildung 4.7 zeigt das verfei-

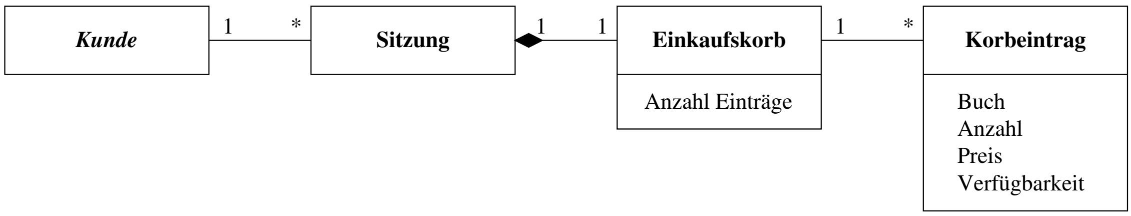


Abbildung 4.7: Verfeinerung des Fachklassendiagramms zum Einkaufskorb mit einer Kompositionsbeziehung

nerte Fachklassendiagramm zum Einkaufskorb. Die Assoziation zwischen den Fachklassen Sitzung und Einkaufskorb wurde als Komposition verfeinert.

Als Besonderheit bei der Sitzungs–Klasse erweist sich, daß sich diese nicht aus der Beschreibung im Pflichtenheft ableitet, sondern bei der Modellierung des Fachklassendiagramms notwendig wurde. Es stellt die Verbindung zwischen Kunden und Einkaufskorb her und verwaltet wie oben beschrieben den Einkaufskorb. Im Abschnitt Produktdaten des Pflichtenheftes sind hierzu keine speziellen Angaben zu finden.

4.1.3 Bestellungen

Aus den im Pflichtenheft beschriebenen Bestelldaten (siehe */D:Bestellungen/* im Abschnitt 3.4 auf Seite 15) wird in diesem Abschnitt das Fachklassendiagramm erstellt. Im ersten Teil dieses Abschnitts wird die Bestellung mit den Bestelleinträgen inklusive Kunden und Bücher modelliert. Der zweite Teil beschäftigt sich mit Rechnungen, Mahnungen und Zahlungsweisen.

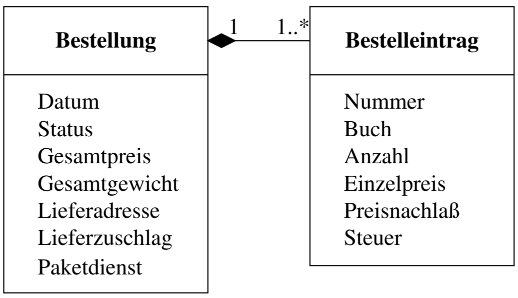


Abbildung 4.8: 1. Fachklassendiagramm zur Bestellung

Eine Bestellung besteht aus mehreren Bestelleinträgen, und zwischen einem Bestellungsobjekt und einem Bestelleintragsobjekt besteht eine Kompositionsbeziehung. Diese Beziehung liegt vor, da die Existenz eines Bestelleintragsobjekts von der eines Bestellobjekts abhängt, das Bestelleintragsobjekt über das Bestellungsobjekt angesprochen wird und ein Bestelleintrag genau zu einer Bestellung gehört.

Ein Bestelleintrag in einer Bestellung enthält zur Identifizierung, z.B. auf einer Rechnung, eine laufende Nummer. Da auch die abgeschlossenen Bestellungen zur Berechnung von Statistiken und zur Ermittlung des Gesamtbestellvolumens einer Person erhalten bleiben, wird im Attribut Status die Abarbeitung der jeweiligen Bestellung vermerkt. Eine Definition der möglichen Status–Werte ist in Abbildung 4.22 auf Seite 40 zu finden.

Ein Lieferzuschlag kann z.B. für Auslandslieferungen erforderlich sein und ein Preisnachlaß kann bei großen Stückzahlen oder bei häufigen Bestellungen eingeräumt werden.

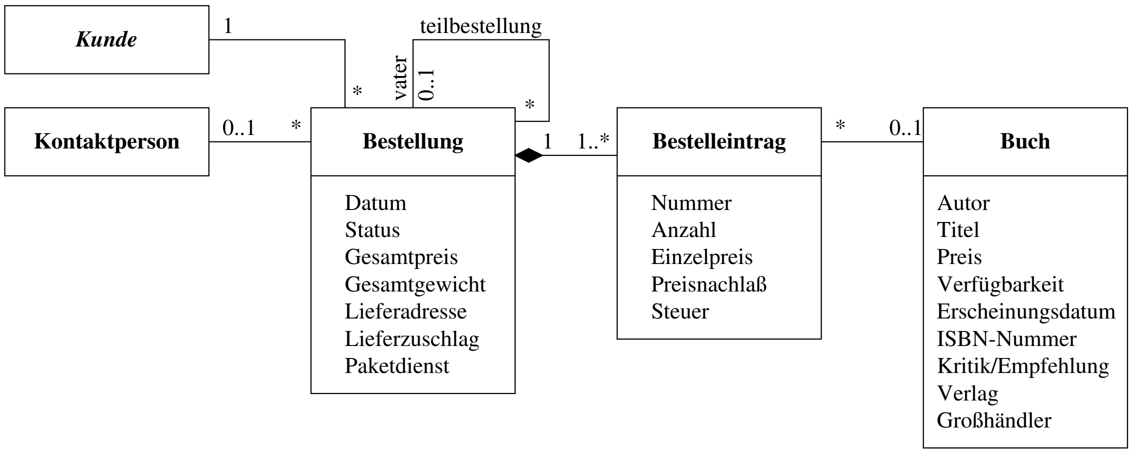


Abbildung 4.9: 2. Fachklassendiagramm zur Bestellung

Da es jedoch z.B. bei größeren Lieferungen oder bei spezieller Fachliteratur zu Verzögerungen bei der Buchbeschaffung kommen kann, wird es notwendig, daß Teile einer Bestellung sofort abgearbeitet werden, während die restlichen Bestelleinträge erst zu einem späteren Termin geliefert werden können. Zur Realisierung dieser Forderung werden Teilbestellungen eingeführt,was zu einer rekursiven Struktur führt, wie dies in Abbildung 4.9 zu sehen ist. Einer Bestellung können nun $0 \ldots n (\equiv *)$ Teilbestellungen zugeordnet werden und eine Teilbestellung gehört zu einer übergeordneten Bestellung bzw. enthält $0 \ldots n$ Teilbestellungen. Eine Ausnahme bildet dabei die oberste Bestellung, die keinen Vater haben kann, was zur Folge hat, daß die Vielfachheit der Vaterbestellung neben der Eins auch eine Null zulassen muß.

Eine Diskussion zur detaillierten Spezifikation der rekursiven Teilbestellung mit Hilfe der OCL findet in dem Abschnitt 4.2: „Designentscheidung zur rekursiven Teilbestellung“ auf Seite 34 statt.

Neben der Einführung der rekursiven Teilbestellung wurden auch die Assoziationen zu Kunden, Kontaktpersonen und Büchern im Diagramm dargestellt. Ein Kunde kann mehrere Bestellungen aufgeben oder es können mehrere, bereits abgearbeitete Bestellungen zu einem Kunden in der Bestellungsverwaltung gespeichert sein. Eine Assoziation mit einer Kontaktperson wird nur bei Firmenkunden nötig, denn neben den Firmenangaben muß auch ein Bezug zu einem Bearbeiter möglich sein. Auch dieser Eigenschaft muß eine Invariante eingehalten werden, die auch im Abschnitt 4.2 auf Seite 34 erläutert und eingeführt wird.

Da auch alte Bestelleinträge, d.h. von früheren Bestellungen, innerhalb des Systems gespeichert werden, kann es z.B. vorkommen, daß ein Buch, welches von einem Bestelleintrag referenziert wird, bereits aus dem Buchkatalog entfernt wurde. Aus diesem Grund wurde die Vielfachheit bei der Assoziation Bestelleintrag – Buch auf der Buchseite auf `0..1` gesetzt, so daß ein Bestelleintrag auch ohne zugeordnetes Buch weiter existieren kann. Die hier dargestellte Fachklasse Buch wird erst im nächsten Teilabschnitt zum Buchkatalog (siehe Teilabschnitt 4.1.4 auf Seite 28) eingeführt. Sie wurde hier vorab dargestellt, wobei noch nicht alle Attribute aufgeführt sind.

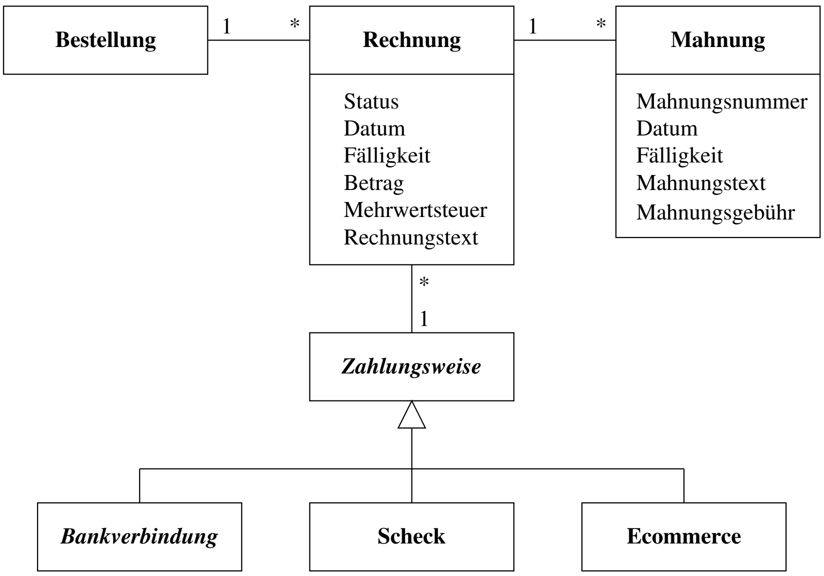


Abbildung 4.10: 3. Fachklassendiagramm zur Bestellung

Orthogonal zu den bereits diskutierten Kunden–, Kontaktperson– und Bestelleintrags–Assoziationen folgt nun die Zuordnung der Rechnung zum Bestellobjekt. Einer Bestellung können mehrere Rechnungen zugeordnet werden. Diese Forderung ergibt sich z.B. dadurch, daß ein Kunde nicht den vollen Rechnungsbetrag begleicht und eine zusätzliche Rechnung zum Einfordern des Restbetrags erstellt werden muß. Im Attribut „Status“ wird die Begleichung der Rechnung vermerkt. Sollte eine Rechnung nicht bezahlt werden, so können zu ihr mehrere Mahnungen erstellt werden.

Eine Rechnung wird immer durch eine Zahlungsweise beglichen. Zur Verfügung stehen hierzu die Bankverbindung, die bereits im Abschnitt 4.1.1 auf Seite 18 diskutiert wurde, eine

Scheckbezahlung und eine E–Commerce–Bezahlungsabwicklung über das Internet. Modelliert werden die verschiedenen Zahlungsabwicklungen durch eine Vererbungshierarchie mit der Superklasse Zahlungsweise und den Subklassen Bankverbindung, Scheck und E–Commerce. Unter der Subklasse Bankverbindung verbergen sich noch die Subklassen Bankeinzug und Kreditkarte. Zwischen den Klassen existiert eine IS–A–KIND–OF–Beziehung, d.h. die Zahlungsweise wird zur Laufzeit mit verschiedenen Objekten der Subklassen instantiiert.

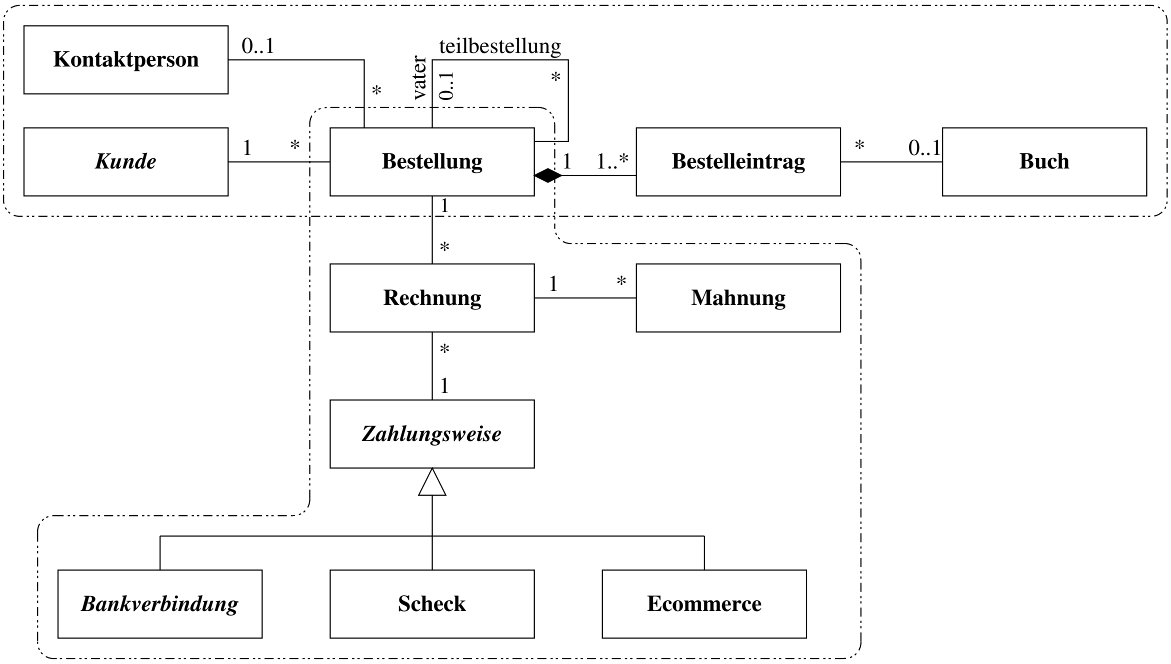


Abbildung 4.11: 4. Fachklassendiagramm zur Bestellung

Abbildung 4.11 zeigt das partielle Fachklassendiagramm zu den Bestellungen nach der Zusammenführung der schrittweise entwickelten Teil–Fachklassendiagramme[2]. Da die Teil–Fachklassendiagramme orthogonal entwickelt wurden, ergaben sich bei der Zusammenführung keine weiteren Assoziationen zwischen den Teildiagrammen.

Das erstellte Fachklassendiagramm bietet in diesem Stadium keinen Anlaß zur Anwendung von Verfeinerungstechniken, da die notwendigen Assoziationen bereits angelegt und ein „Subclassing“ bereits bei der Entwicklung des Teil–Fachklassendiagramms zur Zahlungsweise angewendet wurde.

[2]Die strichpunktierten Linien kennzeichnen die vorher entwickelten Teil–Fachklassendiagramme.

4.1.4 Buchkatalog

Die buchspezifischen Daten vom Autor über Preis bis zur ISBN–Nummer werden neben den bestellspezifischen Daten wie Lagerbestand und Einkaufspreis im Buchkatalog abgelegt. Eine kurze Beschreibung der Buchkatalogdaten wurde bereits im Pflichtenheft unter dem Punkt */D:Buchkatalog/* auf Seite 15 gegeben. In diesem Teilabschnitt erfolgt nun die Umsetzung der verbalen Produktdatenbeschreibung zum Buchkatalog in das Fachklassendiagramm zum Buchkatalog.

Abbildung 4.12: 1. Fachklassendiagramm zum Buchkatalog

Ausgangspunkt für die Diskussion des Buchkatalogs ist das in Abbildung 4.12 dargestellte Fachklassendiagramm. Hierbei wurden die zur Beschreibung, Lagerung und Bestellung notwendigen Daten eines Buches in die Klasse Buch aufgenommen. Der Buchkatalog verwaltet eine Liste von Büchern, die unstrukturiert im System gehalten werden, und das Änderungsdatum sowie die Anzahl der Bücher im Katalog. Die Buchanzahl kann somit bei statistischen Berechnungen ohne vollständiges Durchlaufen des Buchkatalogs ermittelt werden. Das Änderungsdatum dient vor allem zur Überprüfung von Zwischenspeichern (Caches) z.B. beim Kunden. Das System überprüft das Änderungsdatum der Bücher im Cache mit dem des Buchkatalogs. Der Zwischenspeicher muß nur dann neu geladen werden, wenn aktuelle Änderungen im Katalog eingebracht wurden.

Sinnvoll wäre eine Untergliederung des Buchkatalogs in Sachbereiche. Will z.B. ein Kunde im Katalog blättern, so kann er sich durch die einzelnen Sachbereiche bewegen, ohne daß durch eine vorgeschaltete Suche der ganze Buchkatalog durchsucht und die entsprechenden Bücher in einer Liste gesammelt werden. Außerdem würde dann die zu einem riesigen Bestand anwachsende Büchermenge strukturiert im System abgelegt werden können. Eine konkrete Suche müßte dann nicht mehr alle Einträge im Buchkatalog durchsuchen, sondern könnte sich auf einzelne Sachbereiche beschränken.

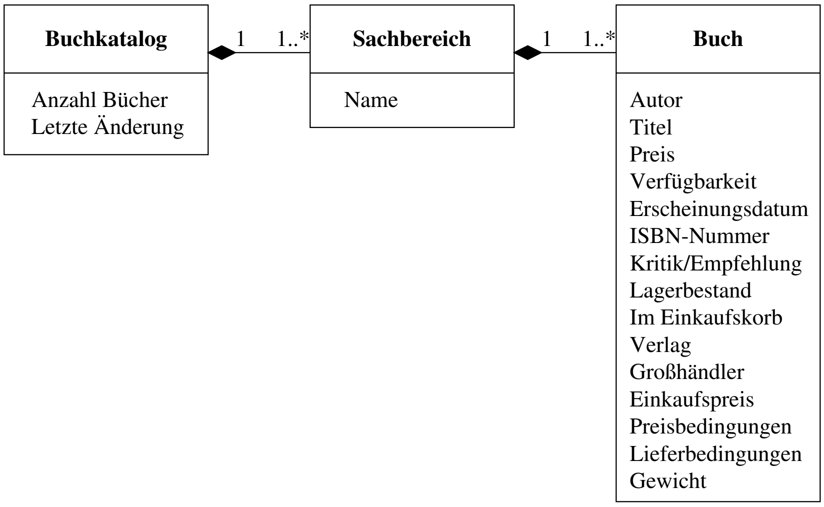


Abbildung 4.13: 2. Fachklassendiagramm zum Buchkatalog

Abbildung 4.13 zeigt das Fachklassendiagramm zum Buchkatalog nach dem Ergänzen der Klasse Sachbereich. Das Attribut Sachbereich aus der Klasse Buch wurde nun in eine Komposition transformiert. Zwischen Buchkatalog und Sachbereich sowie zwischen Sachbereich und Buch existiert jeweils eine Kompositionsbeziehung, da die Lebenszeit der Buchobjekte vom Sachbereichsobjekt und dessen Lebenszeit wiederum vom Buchkatalogobjekt abhängt. Weiterhin gehört ein Buch einem Sachbereich und ein Sachbereich genau einem Buchkatalog an.

Denkbar wäre auch eine rekursive Strukturierung der Sachbereiche. So könnten weitere Unterteilungen des Buchkatalogs erreicht werden. In diesem Fall müßte allerdings aus Gründen der Handhabbarkeit auch die Möglichkeit „einfacher“ Suchanfragen, d.h. Anfragen in denen die Sachbereichshierarchie vor dem Benutzer verborgen bleibt, beibehalten werden. Diese Verfeinerung wird hier nicht weiter ausgeführt. Der Buchkatalog kann durch eine Definition von weiteren Sachbereichen in genügend kleine Teile aufgegliedert werden, so daß keine weiteren Untergliederungen notwendig sind.

Das Attribut Großhändler in der Klasse Buch erlaubt die Angabe eines Vertriebspartners, über den das Buch bestellt und zum Bookshop geliefert werden kann. Da jedoch auch die Möglichkeit besteht, daß die Verlage, welche die Bücher produzieren, eine eigene Spedition haben und aus diesem Grund ihre Bücher selbst ausliefern, wird das Attribut Großhändler zu Vertrieb umbenannt. Die Klasse Vertrieb, die nun die gemeinsamen Daten und Eigenschaften von Verlagslieferungen und Großhändlerlieferungen inne hat, kann nun von der Klasse Buch referenziert werden.

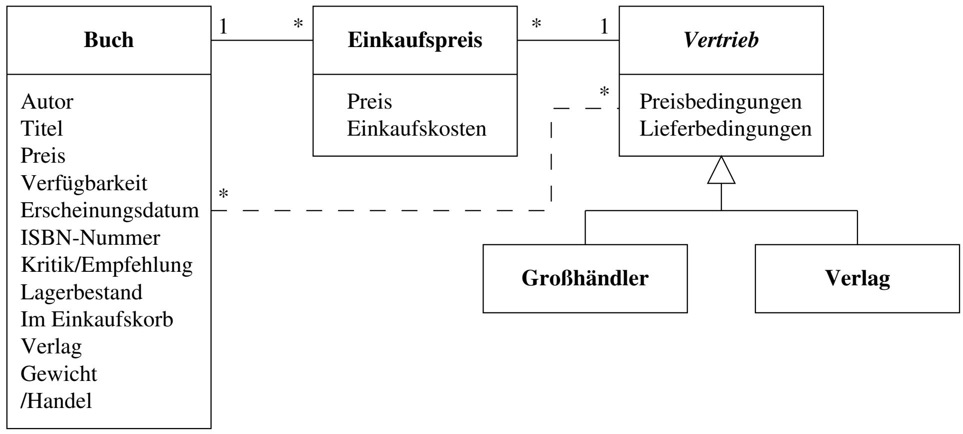


Abbildung 4.14: 3. Fachklassendiagramm zum Buchkatalog

Da jedoch verschiedene Großhändler oder die Verlage selbst zu verschiedenen Preisen liefern können und diese Lieferpreise auch den Buchpreis beeinflussen, wird zwischen der Klasse Buch und der Klasse Vertrieb die Klasse Einkaufspreis eingeführt. Diese Struktur ermöglicht es, daß zu einem Buch verschiedene Einkaufspreise bezüglich der verschiedenen Vertriebe gespeichert werden. So kann bei einer Bestellung der billigste Vertrieb ausgewählt und bei ihm bestellt werden. Da oftmals auch die Endverkaufspreise von Büchern festgelegt sind, kann der Bookshop mit dem Attribut Einkaufskosten in der Klasse Einkaufspreis den größten Gewinn in bezug auf den Vertriebspartner ermitteln. Die Transformierung des Buchattributs Großhändler in die Fachklasse Vertrieb mit Ergänzung der Fachklasse Einkaufspreis ist in Abbildung 4.14 zu sehen. Zu beachten ist an dieser Stelle, daß bei dieser Transformation die Eigenschaften der Klasse Buch nicht erhalten werden, da keine direkte Beziehung zum Vertrieb bestehen bleibt. Die Erhaltung der Eigenschaften, bis auf die Isomorphie, wäre möglich durch Definition einer abgeleiteten (derived) Assoziation zwischen der Klasse Buch und der Klasse Vertrieb. Dieser Assoziation müßte dann das Relationenprodukt aus den Assoziationen zwischen Buch und Einkaufspreis sowie Einkaufspreis und Vertrieb zugeordnet werden. Die abgeleitete Assoziation wird in Abbildung 4.14 gestrichelt dargestellt (/Handel).

Eine vollständige Ermittlung des Fachklassendiagramms zu den Vertrieben mit den zugehörigen Attributen findet im nächsten Teilabschnitt (4.1.5: „Vertrieb“ auf Seite 31) statt.

Abbildung 4.15 auf der nächsten Seite zeigt nun noch die Vereinigung der Teildiagramme zum Fachklassendiagramm des Buchkatalogs. Wie auch bei der Vereinigung zum Fachklassendiagramm der Bestellung im Teilabschnitt 4.1.3 auf Seite 24 ergaben sich bei der Zusammenführung keine weiteren Assoziationen, da die Teildiagramme so gewählt wurden, daß sie voneinander unabhängig bearbeitet werden konnten.

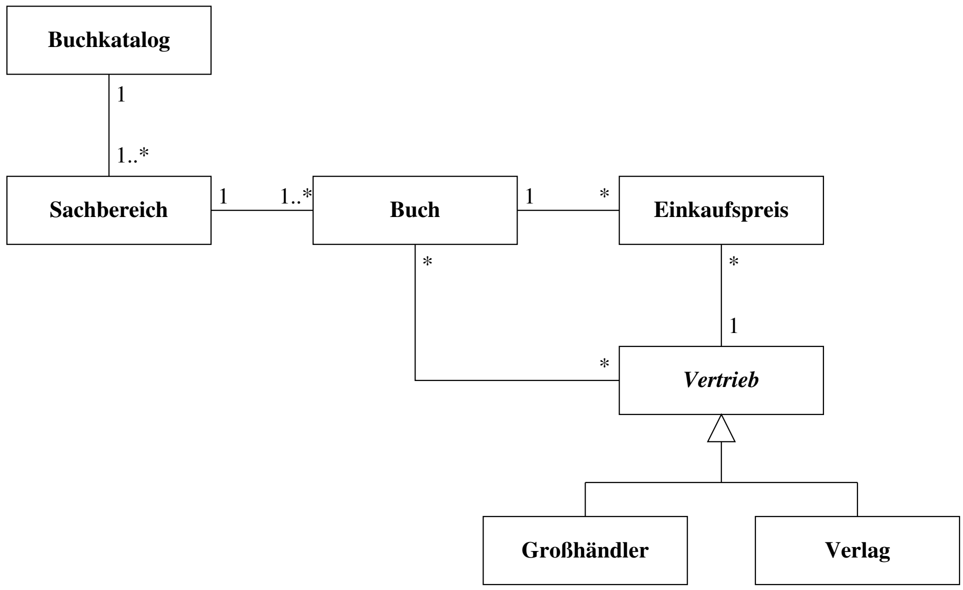


Abbildung 4.15: 4. Fachklassendiagramm zum Buchkatalog

4.1.5 Vertrieb

In der Beschreibung der Produktdaten im Pflichtenheft wurden bereits die Großhändlerdaten erläutert (siehe Abschnitt 3.4 auf Seite 15). Während der Analyse des Anwendungsbereichs hat sich ergeben, daß auch Bücher direkt beim Verlag bestellt und von diesem ausgeliefert werden können (siehe Teilabschnitt 4.1.4 auf Seite 28), so daß neben der Fachklasse Großhändler auch eine Fachklasse Verlag existiert. Das Fachklassendiagramm zu diesen beiden Klassen ist in Abbildung 4.16 zu sehen.

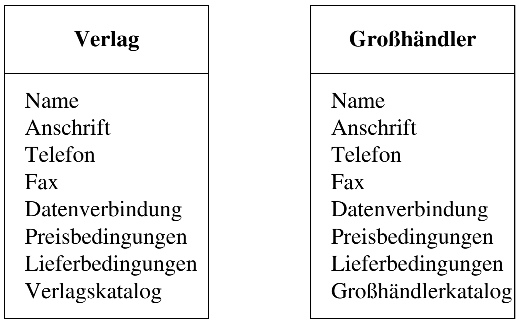


Abbildung 4.16: 1. Fachklassendiagramm zum Vertrieb

Bei einer genaueren Betrachtung der Fachklassen zum Verlag und zum Großhändler ergibt sich, daß beide Klassen dieselben Attribute enthalten. Die Attribute können hier wieder, analog zu den Privat– und Firmenkunden (siehe 4.1.1 auf Seite 18), in einer abstrakten Superklasse zusammengefaßt werden, in der auch später die gemeinsamen Operationen definiert werden. Zu diesem Zeitpunkt kann die Definition der Oberklasse und die Migration der Attribute, wie sie in [OJ93] vorgeschlagen wird, bereits stattfinden. Angewandt wird also wieder die Transformation, die von William Opdyke in [Opd92] als „Refactoring to Generalize: Creating an Abstract Superclass“ bezeichnet wird.

Bei der Vererbungsbeziehung handelt es sich analog zu den Kundenklassen um eine IS–A–KIND–OF–Beziehung. Abbildung 4.17 zeigt das zugehörige Fachklassendiagramm nach der Transformation.

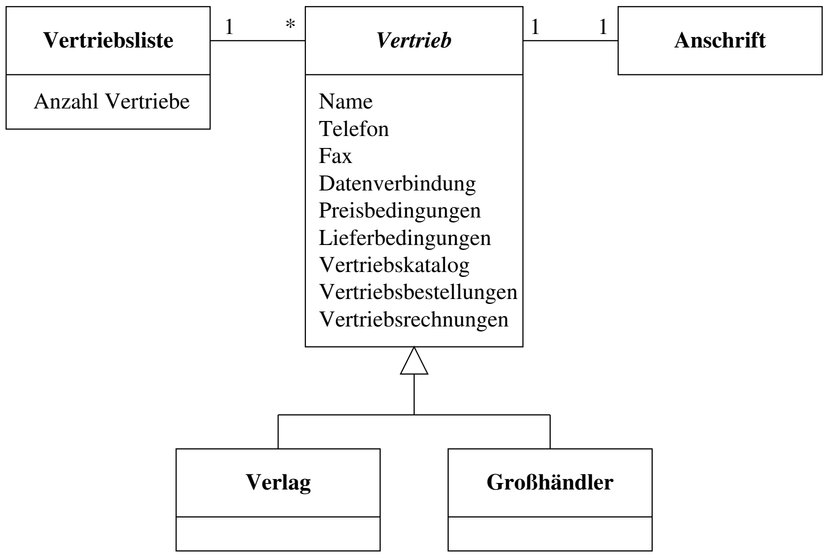


Abbildung 4.17: 2. Fachklassendiagramm zum Vertrieb

Durch die angewandte Transformation wird nun eine Speicherung der Vertriebe, die sowohl als Verlag oder auch als Großhändler ausgeprägt sind, in einer einzigen Vertriebsliste ermöglicht. Das Attribut Anschrift wurde aus den Fachklassen Verlag und Großhändler entfernt und in die Klasse Vertrieb wurde eine Assoziation zur Klasse Anschrift eingefügt, was später bei einer Instantiierung zu der Anwendung eines Referenzobjekts an Stelle eines Wertobjekts führt.

Die Datenverbindung, für die ein eigenes Attribut zur Verfügung gestellt wird, dient dem Informationsaustausch zwischen Bookshop und Vertriebspartner. So können z.B. der Großhändlerkatalog in elektronischer Form (z.B. in einer HTML–oder ASCII–Datei) oder Vertriebsbestellungen ausgetauscht werden. Hinter der Datenverbindung kann sich in der einfachsten Form auch nur eine Email–Adresse verbergen, über welche sich die gesamte Datenkommunikation in vorgegebenen Dateiformaten abspielt.

Da zur Datenverbindung in diesem Entwicklungsstadium noch keine konkrete Realisierungsaussage getroffen wird, bleibt noch offen, ob es sich um ein Wert– oder Referenzattribut handelt. Wir interpretieren also die fehlende Typangabe im Sinne der losen Semantik. Dies gibt uns die Möglichkeit, die Spezifikation in weiteren Entwicklungsschritten zu verfeinern, indem wir noch offene Details dann explizit festlegen. Hier könnte dies durch Angabe eines Werttyps oder durch Einführung einer Assoziation geschehen.

Zu den Fachklassen Verlag und Großhändler existieren zu diesem Zeitpunkt keine speziellen Attribute. Jedoch können sich im Laufe der Bearbeitung spezifische Attribute oder Operationen ergeben, die in der entsprechenden Fachklasse ergänzt werden.

4.1.6 Paketdienste

Aus der Datenbeschreibung zu den Paketdiensten im Pflichtenheft (siehe */D:Paketdienste/* auf Seite 15) ergibt sich schließlich das partielle Fachklassendiagramm zu den Paketdiensten, wie es in Abbildung 4.18 dargestellt ist.

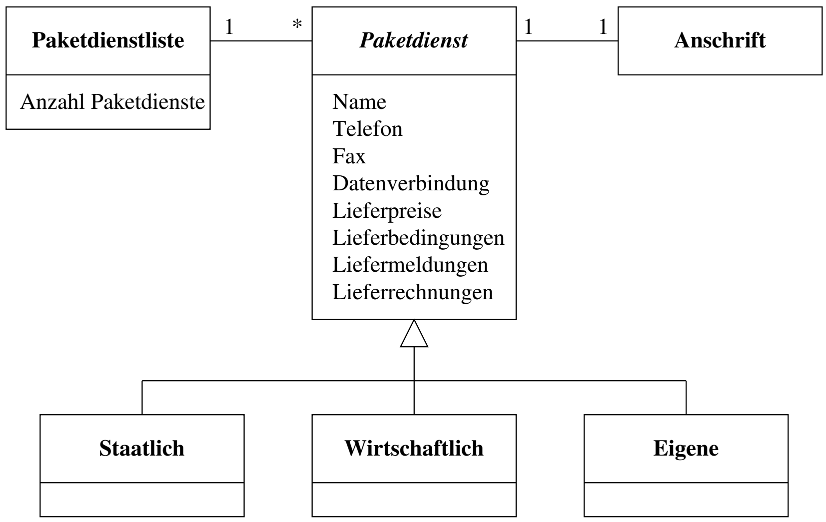


Abbildung 4.18: Fachklassendiagramm zu den Paketdiensten

Die IS–A–KIND–OF–Beziehung zwischen der abstrakten Superklasse Paketdienst und den Subklassen Staatlich, Wirtschaftlich und Eigene erlaubt eine Instantiierung der Subklassen als Objekte vom Typ Paketdienst. Hierdurch können die verschiedenen Paketdienste wieder in einer gemeinsamen Liste verwaltet werden.

Die im Pflichtenheft beschriebenen Auslieferungszeiten, die zu einem Paketdienst festgehalten werden, finden sich hier unter anderem in dem Attribut Lieferbedingungen wieder.

Neben den Vertrieben erhalten auch die Paketdienste eine Datenverbindung, über die z.B. Auslieferungsaufträge und –bestätigungen mitgeteilt werden können.

4.2 Designentscheidung zur rekursiven Teilbestellung

In diesem Abschnitt wird die rekursive Aufteilung von Bestellungen in Teilbestellungen mit den zugehörigen Forderungen ausführlich diskutiert. Eine Einführung zur rekursiven Struktur wird im Teilabschnitt 4.2.1 gegeben. Teilabschnitt 4.2.2 beschreibt allgemeine Invarianten unabhängig vom verwendeten Design–Ansatz. Die Teilabschnitte 4.2.3, 4.2.4 und 4.2.5 befassen sich dann mit den konkreten Ansätzen. Abschließend wird im Teilabschnitt 4.2.6 ein Ergebnis der Untersuchung zusammengefaßt.

Detaillierte Eigenschaften der Teilbestellung lassen sich mit Klassendiagrammen alleine nicht präzise und eindeutig spezifizieren. Diese müssen durch Invarianten ausgedrückt werden, die in einer formalen Sprache formuliert werden. Invarianten formulieren Eigenschaften von Objektstrukturen, die zu jedem Zeitpunkt gelten sollen. Insbesondere schränken sie die Attributwerte von Objekten und deren Bezüge zu anderen Objekten ein oder beschreiben funktionale Abhängigkeiten.

Als formale Sprache wurde hier die OCL eingesetzt, die ebenfalls Teil der UML 1.1 ist. Hierbei handelt es sich nach [Rat97] um eine reine Sprache für Ausdrücke, die keine Seiteneffekte und Veränderungen im Modell hervorrufen. Sie ist auch keine Programmiersprache, sondern eine Modellierungssprache. Außerdem handelt es sich hierbei um eine typisierte Sprache. Erläuterungen zu Invarianten und zur OCL sind in [Rat97], [Bre98a], [Bre98b], [KWC98], [Rum98c] und [BBH$^+$99] zu finden.

4.2.1 Einführung

Bei Bestellungen, die umfangreich sind, die mehrere Exemplare eines Buches anfordern oder die z.B. zwei Bücher beinhalten, von denen nur eines sofort lieferbar ist, teilt sich die Bestellung in Teilbestellungen auf. Dies ermöglicht bei den Teilbestellungen ein sinnvolles Setzen des Bestellstatus, eine Berechnung des Gewichts, eine Änderung der Lieferadresse und eine Wahl des Paketdienstes.

Das Gewicht einer Teilbestellung kann für die Kostenberechnung und –optimierung der Teillieferung von Bedeutung sein, besonders wenn unterschiedliche Paketdienste verschiedene Gewichtseinteilungen bezüglich Lieferentgelt anbieten, so daß zu einer Teillieferung der günstigste Lieferant ermittelt werden kann.

Abbildung 4.19 zeigt das Klassendiagramm zur rekursiven Bestellung. Die rekursive Bestellung führt zu einer Baumstruktur, denn eine Bestellung kann auf $0 \ldots n$ Teilbestellungen verweisen und eine Teilbestellung hat genau eine übergeordnete Bestellung. Eine Ausnahme bildet hier die Wurzel, die als einziger Knoten keinen Vorgänger hat, jedoch aber vorhanden sein muß.

Eine Baumstruktur eignet sich für die Struktur der rekursiven Teilbestellungen, denn soll eine Bestellung bzw. Teilbestellung in weitere Teilbestellungen aufgeteilt werden, müssen

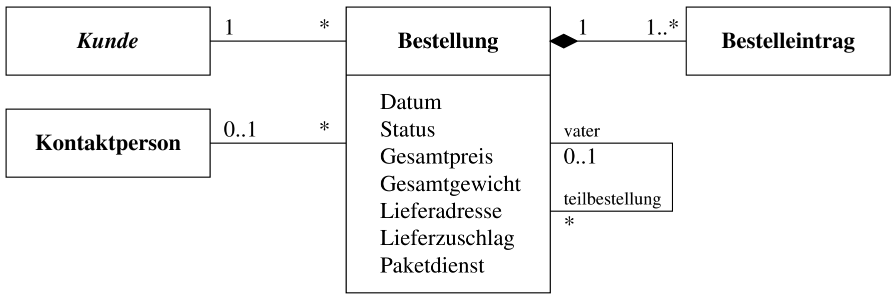


Abbildung 4.19: Fachklassendiagramm zur rekursiven Teilbestellung

bei dem entsprechenden Knoten einfach weitere Knoten als Kinder eingesetzt werden, die dann z.B. die bereits ausgelieferten und die noch auszuliefernden sowie die nicht lieferbaren Bücher als Teilbestellung explizit darstellen. Das ursprüngliche Bestellobjekt bleibt jedoch bestehen. Die Teilbestellungen als Söhne dieses Bestellobjekts müssen zusätzlich erzeugt werden und die Bestelleintragsobjekte migrieren zu den Teilbestellungen.

Da die Teilbestellungsobjekte ebenfalls Objekte vom Typ Bestellung sind, müssen nicht immer alle Attribute in den Objekten innerhalb des Baumes belegt werden. Es können verschiedene Ansätze gewählt werden, die ein redundantes Speichern von Information im Baum ausschließen oder in machen Fällen zumindest reduzieren. Im folgenden werden verschiedene Ansätze dazu diskutiert. Die hierzu notwendigen detaillierten Eigenschaften werden durch Invarianten beschrieben.

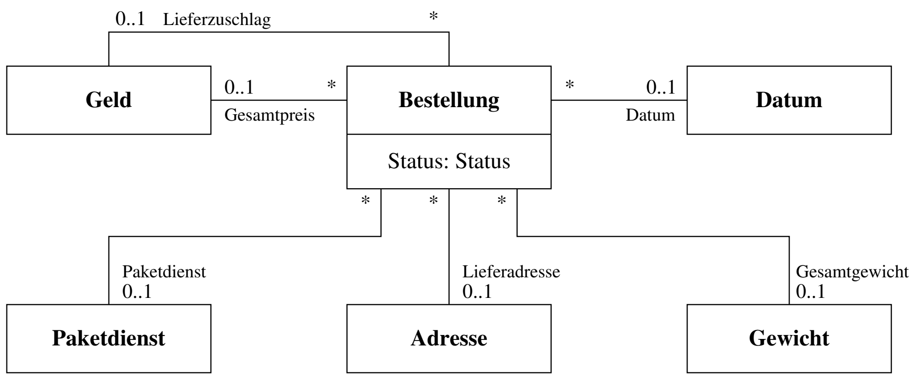


Abbildung 4.20: Typisierung der Referenzattribute im fachlichen Modell der Bestellung

Da diese Invarianten in OCL, einer getypten Spezifikationssprache, formalisiert werden, ist es notwendig, sämtliche betroffenen Attribute explizit zu typisieren. Dies betrifft insbesondere die Attribute der Klasse Bestellung: Datum, Status, Gesamtpreis, Gesamtgewicht, Lieferadresse, Lieferzuschlag und Paketdienst.

Dabei treffen wir auf der hier modellierten fachlichen/konzeptuellen Ebene die folgende Vereinbarung: Alle Attribute der Klasse Bestellung (siehe Abbildung 4.19 auf der vorherigen Seite) außer dem Status–Attribut sind als Referenzattribute zu verstehen, d.h. repräsentieren Assoziationen der Kardinalität 0...1 zu Objekten der jeweiligen Klassen gemäß Abbildung 4.20 auf der vorherigen Seite. Das Status–Attribut wird in Abbildung 4.22 auf Seite 40 als Aufzähltyp definiert.

Wie schon bei der Diskussion der Bestellung erläutert wurde, muß bei einem Firmenkunden auch die Kontaktperson innerhalb einer Bestellung identifiziert werden, bei Privatkunden bleibt dieses Attribut unbelegt. In den folgenden Diskussionen der Designentscheidung wird vereinfachend nur von Kundenreferenzen gesprochen, ohne zwischen Privat– und Firmenkunden zu unterscheiden. Im Falle einer Firmenbestellung verbirgt sich hinter dem Attribut Kunde auch gleichermaßen das Attribut Kontaktperson, für das die selben Einschränkungen gelten wie für die Kundenreferenz.

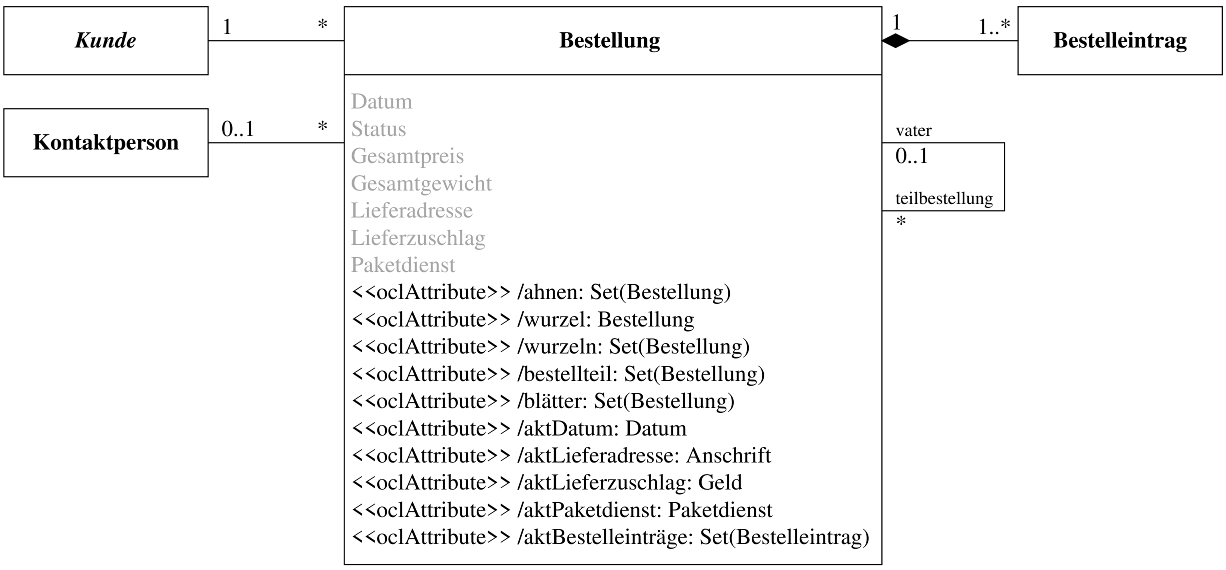


Abbildung 4.21: Fachklassendiagramm zur rekursiven Teilbestellung mit Erweiterung der abgeleiteten Attribute

Da in den folgenden Invarianten oftmals Hilfsvariablen zur besseren Lesbarkeit eingeführt werden, ergänzen wir diese in der Klasse Bestellung. Da es sich dabei aber um Attribute handelt, die mit der Hilfe von anderen Attributen und Assoziationen berechnet werden können, handelt es sich dabei um sogenannte abgeleitete Attribute (engl. „derived attributes“, vgl. [FS98]). Abbildung 4.21 zeigt das um die abgeleiteten Attribute erweiterte Klassendiagramm zur rekursiven Teilbestellung. Die abgeleiteten Attribute werden mit einem vorangestellten Schrägstrich dargestellt. Da sie nur verwendet werden, um Invarianten in OCL zu spezifizieren, aber in einer möglichen Implementierung des Systems nicht vorkommen, werden sie mit dem Stereotyp <<OCLATTRIBUTE>> gekennzeichnet.

4.2.2 Allgemeine Invarianten

In diesem Teilabschnitt werden Invarianten definiert, die unabhängig von einem gewählten Ansatz beim Bestellungsobjekt immer gelten müssen. So muß z.B. bei einer Firmenkundenbestellung eine Kontaktperson mit angegeben werden. Dieser Sachverhalt mit der folgenden Invariante ausgedrückt.

```
Bestellung
kunde.OclIsTypeOf(Firmenkunde) = (kontaktperson->notEmpty)
```

Ist der referenzierte Kunde ein Firmenkunde, so muß die Kontaktpersonenreferenz belegt sein. Andernfalls, also bei einem Privatkunden, darf sie nicht belegt sein.

Unabhängig vom verwendeten Ansatz muß die rekursive Struktur der Bestellungen einer Baumstruktur entsprechen. Die folgenden Invarianten legen die Eigenschaften der Baumstruktur fest, u.a. die Zyklenfreiheit.

1. Im Baum muß genau ein Wurzelknoten existieren, d.h. es gibt einen Knoten, der keinen Vorgänger besitzt.

   Zuerst wird hierzu die Menge „bestellteil" gebildet, die zu einem Bestellknoten alle nachfolgenden Knoten in der Baumstruktur liefert. Zusätzlich wird der Startknoten ebenfalls in diese Menge aufgenommen.

   ```
   Bestellung
   bestellteil = Set{self}->union(teilbestellung.bestellteil)
   ```

   Bei der gerade definierten Gleichung handelt es sich um eine rekursive Gleichung, denn die Menge „bestellteil" kommt sowohl auf der rechten wie auf der linken Seite der Gleichung vor. Bei der Bestimmung der Lösungsmenge ist die mengentheoretisch kleinste Lösung von Interesse, d.h. es werden ausschließlich diejenigen Objekte in die Lösungsmenge mitaufgenommen, die über die Vater–Beziehung erreichbar sind[3].

   Anschließend wird die Menge der Ahnen (Vorgänger) zu einem Knoten gebildet. Aus allen Bestellknoten werden diejenigen Knoten ausgewählt, in deren Nachfolgermenge („bestellteil") sich der gegebene Knoten befindet.

   ```
   Bestellung
   ahnen = Bestellung->allInstances->
             select(bestellteil->includes(self))->excluding(self)
   ```

[3]Faßt man den Ausdruck auf der rechten Seite als Abbildung mit dem Parameter „bestellteil" auf, so interessieren wir uns für den kleinsten Fixpunkt der Abbildung (vgl. [Bro92])

Unter den Ahnen befindet sich dann ein Knoten, der keinen Vorgänger (Vater) besitzt. Ein solcher Knoten kommt genau einmal vor. Dieser Wurzelknoten („wurzel“) wird in weiteren Invarianten benötigt und deshalb als abgeleitetes Attribut gespeichert.

```
Bestellung
wurzeln = ahnen->select(vater->isEmpty) and
wurzeln->size = 1
```

```
Bestellung
wurzeln->includes(wurzel)
```

2. Nun müssen für den folgenden Sachverhalt die Invarianten erstellt werden: In dem Baum einer rekursiven Bestellung muß es mindestens einen Blattknoten geben. Da bei einer Aufteilung einer Bestellung die Bestelleinträge immer in die Teilbestellungen abwandern, gelangen die Bestelleinträge immer in die Blattknoten. Da jedoch einer Bestellung immer mindestens ein Bestelleintrag zugeordnet wird, muß sich zumindest dieser in einem Blatt befinden. Bei einem einzelnen Bestelleintrag ist der Wurzelknoten ebenfalls Blattknoten.

   Hierzu wird zuerst die Menge aller Blattknoten im Baum ermittelt. Aus der Nachflogermenge („bestellteil“) des Wurzelkotens werden diejenigen Knoten ausgewählt, denen keine Teilbestellungen zugeordnet sind. Es muß mindestens ein solcher Knoten existieren.

```
Bestellung
blätter = wurzel.bestellteil->select(teilbestellung->isEmpty) and
blätter->size > 0
```

3. Die Vielfachheiten zur rekursiven Teilbestellung geben an, daß zu einem Bestellobjekt kein oder ein Vaterobjekt und beliebig viele Teilbestellungsobjekte existieren.

   Hierzu muß folgende Forderung erfüllt sein. Falls der aktuelle Bestellknoten einen Vater hat, so muß er ein Element der Menge der Teilbestellungen seines Vaters sein[4].

   Dieses Constraint könnte auch weggelassen werden, da dieser Sachverhalt auch durch die Assoziation „vater“ – „teilbestellung“ bereits gesichert ist.

```
Bestellung
vater->notEmpty implies vater.teilbestellung->includes(self)
```

[4]Vergleiche hierzu auch das COMPOSITE(163)–Design Pattern in [GHJV98], Abschnitt Implementation, Explicit parent references.

Die Attribute Status, Gesamtpreis und Gesamtgewicht können in jedem Knoten verschieden sein, denn sie enthalten eine Summe bzw. den Gesamtstatus des Teilbaums. Eine Speicherung der Attribute in den Knoten ermöglicht eine schnellere Ermittlung der Gesamtwerte einer Bestellung. Deshalb gelten die folgenden Invarianten unabhängig vom gewählten Ansatz immer.

1. Das Attribut Gesamtgewicht in einer Bestellung setzt sich zusammen aus dem Gewicht der Bestelleinträge[5].

   ```
   Bestellung
   Gesamtgewicht.value = bestellteil.bestelleintrag.Gewicht.value->sum
   ```

2. Das Attribut Gesamtpreis setzt sich aus dem Preis der Bestelleinträge zusammen[6].

   ```
   Bestellung
   Gesamtpreis.value = bestellteil.bestelleintrag.Preis.value->sum
   ```

   Der Preis eines Bestelleintrags ergibt sich aus der Anzahl, dem Einzelpreis, dem Preisnachlaß und der Steuer. Hierzu könnte in der Klasse Bestelleintrag ein abgeleitetes Attribut Preis eingeführt werden. Diese Einführung wird jedoch im Rahmen dieser Arbeit nicht gezeigt.

3. Statusberechnung. Das Statusattribut gibt Auskunft darüber, inwieweit die Bestellung bereits abgearbeitet wurde. Abbildung 4.22 auf der nächsten Seite zeigt die Fachklasse Bestellung mit dem ordinalen Typ Status und seinen möglichen Werten.

   Die Belegung des Statusattributs in übergeordneten Bestellungen berechnet sich aus den Statuswerten der Teilbestellungen. Die Definition der Statuswerte und der Invariante, die zur Bestimmung der Statuswerte verwendet wird, findet im folgenden statt.

   Die Attributwerte haben im Zusammenhang mit der Abarbeitung der Bestellung folgende Bedeutung:

   neu: Nach dem definitiven Bestellen der im Einkaufskorb liegenden Bücher oder nach dem manuellen Hinzufügen von Bestellungen durch das Händlerpersonal werden neue Bestellobjekte erzeugt, bei denen das Attribut Status auf den Wert „neu“ gesetzt wurde. Zu einer offenen Bestellung dürfen keine weiteren Einträge hinzugefügt werden. Dies ist nur so lange möglich, wie die Bestellung selbst den Status „neu“ besitzt. Weitere Bestelleinträge werden durch eine neue Bestellung ermöglicht.

[5] Wir sehen hier voraus, daß die Klasse Gewicht ein skalares Attribut „value“ besitzt.
[6] Wir sehen hier voraus, daß die Klasse Geld ein skalares Attribut „value“ besitzt.

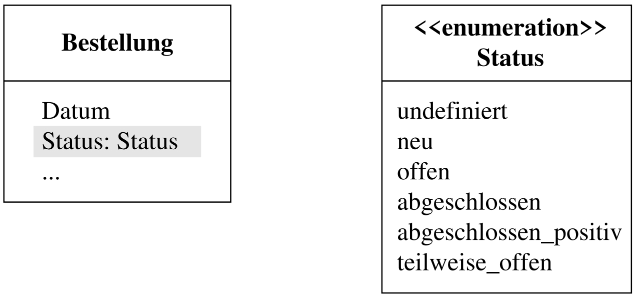


Abbildung 4.22: Definition des ordinalen Status–Typs

offen: Werden die Bestellungen aktiviert und eventuelle Nachbestellungen vom Vertriebspartner angefordert, lautet der Bestellstatus „offen".

teilweise_offen: Diesen Status erhält eine übergeordnete Bestellung dann, wenn Teilbestellungen bereits abgeschlossen und andere noch offen sind.

abgeschlossen_positiv: Wurden von einer Bestellung alle Bestelleinträge bzw. alle Teilbestellungen ausgeliefert, findet ein Update des Status auf diesen Wert statt.

abgeschlossen: Dieser Wert wird eingetragen, wenn entweder ein Bestelleintrag nicht geliefert werden konnte oder bei übergeordneten Bestellungen falls eine Teilbestellung existiert, die den Status „abgeschlossen" enthält.

```
Bestellung
teilbestellung->notEmpty implies status =
  if teilbestellung->exists(Status->isEmpty)
  then #undefined
  else
    if teilbestellung->exists(Status = #neu)
    then #neu
    else
      -- forall teilbestellung.Status <> #neu
      if teilbestellung->forAll(Status = #offen)
      then #offen
      else
        -- exists teilbestellung.Status <> #offen
        if teilbestellung->
          exists(Status = #offen or Status = #teilweise_offen)
        then #teilweise_offen
        else
          -- forall teilbestellung.Status <> #offen and
          --        teilbestellung.Status <> #teilweise offen
          if (teilbestellung->
            forAll(Status = #abgeschlossen_positiv)
```

```
          then #abgeschlossen_positiv
          else
            -- forall teilbestellung.Status <> #abgeschlossen_positiv
            if teilbestellung->exists(Status = #abgeschlossen)
            then #abgeschlossen
            else #undefined
            endif
          endif
        endif
      endif
    endif
  endif
```

Da der Status nur bei übergeordneten Bestellungen, d.h. bei Nicht–Blattknoten, berechnet werden muß, werden nur diejenigen Knoten betrachtet, bei denen keine Teilbestellungen eingetragen wurden. Im Gegensatz dazu wird bei Blattknoten der Status beim Bearbeiten der Bestellungen und Bestelleinträge gesetzt bzw. verändert.

Falls in irgendeiner Teilbestellung der Status nicht gesetzt wurde, kann auch in den übergeordneten Bestellungen keiner gesetzt werden. Dieser Ausnahmefall wird mit der Zuweisung des Wertes „undefined“ gekennzeichnet.

Existiert eine Teilbestellung mit dem Status „neu“, so gilt der „neue“ Bestellstatus auch für die übergeordnete Bestellung (siehe hierzu die obige Beschreibung zum Statuswert „neu“ auf Seite 39).

Sind alle Teilbestellungen offen, dann gilt auch für die Vaterbestellung der Zustand „offen“. Sind dagegen bereits Teile abgeschlossen, dann wird der Wert „teilweise_offen“ zugewiesen.

Konnten alle Bestellungen erfüllt werden, wird das Statusattribut mit „abgeschlossen_positiv“ belegt. Existieren Teilbestellungen, mit nicht lieferbaren Posten, so wird der Wert „abgeschlossen“ angenommen.

Neben diesen allgemeinen Invarianten müssen abhängig vom gewählten Ansatz weitere Invarianten erfüllt sein, die eine sinnvolle und richtige Belegung der Attribute gewährleisten. Diese Diskussion findet in den nächsten drei Abschnitten statt, da sie für die verschiedenen Varianten unterschiedlich sind.

4.2.3 Eintrag in jedem Blatt (Bottom–Ansatz)

In diesem Ansatz enthalten nur die Blätter Informationen über eine Bestellung bzw. Teilbestellung, d.h. die Bestelleinträge, das Bestelldatum, die Lieferadresse, der Lieferzuschlag und der Paketdienst werden auf unterster Ebene vermerkt. Dies führt teilweise zu einer redundanten Speicherung dieser Werte, denn oftmals bleiben Datum, Lieferadresse, –zuschlag und Paketdienst auch in den Teilbestellungen gleich.

Der Status, Gesamtpreis und das Gesamtgewicht werden ebenfalls in den Blättern vermerkt. Diese Informationen sind hier aber von der eigentlichen Teillieferung abhängig und es ergeben sich hierbei keine Redundanzen. Außerdem werden in den Nicht–Blattknoten diese Attribute ebenfalls besetzt, denn dadurch kann das Gesamtgewicht einer Bestellung, der Gesamtpreis und der Status ermittelt werden. Mit Hilfe des Attributs Status kann später überprüft werden, welche Bestellungen bereits abgeschlossen sind und die Attribute Preis und Gewicht werden für Statistikberechnungen benötigt.

Eine Ausnahme bildet die Referenz zum Kundenobjekt. Damit ein umständliches und ineffizienten Suchen nach dem Kunden vermieden wird, erfolgt ein Setzen der Kundenreferenz in jedem Knoten. Beim Erstellen einer Teilbestellung wird diese Referenz miteingetragen. Da eine Bestellung inklusive aller Teilbestellungen immer einem einzigen Kunden zugeordnet ist, ergibt sich durch diese Ausnahmebehandlung kein zusätzlicher Bearbeitungsaufwand.

Die zur Realisierung notwendigen Definitionen der Invarianten für den Bottom–Ansatz und Erklärungen hierzu werden im folgenden aufgelistet.

1. Handelt es sich bei einem Knoten um einen Blattknoten, d.h. falls dieser keine Kinder besitzt, müssen Datum, Status, Gesamtpreis, Gesamtgewicht, Lieferadresse, Lieferzuschlag und Paketdienst eingetragen werden.

```
Bestellung
teilbestellung->isEmpty implies
  Datum->notEmpty and
  Status <> #undefined and
  Gesamtpreis->notEmpty and
  Gesamtgewicht->notEmpty and
  Lieferadresse->notEmpty and
  Lieferzuschlag->notEmpty and
  Paketdienst->notEmpty
```

2. Bei einem „Nicht–Blattknoten“ sind Datum, Lieferadresse, Lieferzuschlag und Paketdienst leer.

```
Bestellung
teilbestellung->notEmpty implies
  Datum->isEmpty and
  Lieferadresse->isEmpty  and
  Lieferzuschlag->isEmpty and
  Paketdienst->isEmpty nil
```

3. Bestelleinträge werden nur in den Blättern eingetragen. Im Fall, daß die Bestellung keine Teilbestellungen enthält, wird der Wurzelknoten auch zum Blattknoten.

```
Bestellung
(teilbestellung->isEmpty implies bestelleintrag->notEmpty) and
(teilbestellung->notEmpty implies bestelleintrag->isEmpty)
```

Die hier eingeführte Invariante, die nur Bestelleinträge bei Blattknoten zuläßt, erfordert eine Änderung im Klassendiagramm der Bestellung. Vor der Invarianten–Definition wurde angenommen, daß einer Bestellung bzw. Teilbestellung mindestens ein Bestelleintrag zugeordnet ist. Die Vielfachheit muß nun auf beliebig viele Bestelleinträge abgeändert werden.

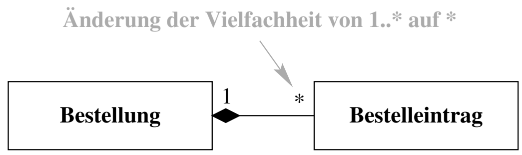


Abbildung 4.23: Änderung der Vielfachheit des Bestelleintrags

Abbildung 4.23 zeigt den Ausschnitt des Fachklassendiagramms nach der Änderung der Vielfachheit. Sie wurde von `1..*` auf `*` geändert.

4.2.4 Eintrag in der Wurzel (Top–Design)

Im Gegensatz zum oben beschriebenen Bottom–Ansatz werden hier beim Top–Design alle Attribute in der Wurzel belegt. Falls sich im Bearbeitungsablauf Teilbestellungen ergeben, werden in diesen nur die abweichenden Daten belegt. Existieren bereits abweichende Daten, so müssen diese in eventuell neu anzulegenden Teilbestellungen ebenfalls gesetzt werden, da beim Feststellen der Daten diese entweder im Blattknoten gesucht werden, oder, falls diese keine Auskunft geben, sofort aus dem Wurzelknoten ermittelt werden.

Die Attribute Status, Gesamtpreis und Gesamtgewicht werden wieder allgemein berechnet und in jedem Knoten abgelegt (siehe Teilabschnitt 4.2.2 auf Seite 37). Die Kundenreferenz wird, wie im vorigen Abschnitt (4.2.3) beschrieben, immer ergänzt.

Die zur Realisierung dieses Ansatzes notwendigen Invarianten werden im folgenden aufgelistet. Die allgemeinen Invarianten (Teilabschnitt 4.2.2 auf Seite 37) gelten zusätzlich.

1. Im Wurzelknoten müssen alle Attribute belegt sein. Wird in den Blattknoten nichts anderes definiert, erfolgt eine Anwendung dieser Werte.

   ```
   Bestellung
   vater->isEmpty implies
     Datum->notEmpty and
     Status <> #undefined and
     Gesamtpreis->notEmpty and
     Gesamtgewicht->notEmpty and
     Lieferadresse->notEmpty and
     Lieferstatus->notEmpty and
     Paketdienst->notEmpty
   ```

   Die Bestelleinträge werden immer in den Blattknoten definiert und müssen deshalb hier nicht gesetzt sein. In dem Fall, daß keine Teilbestellungen existieren und der Wurzelknoten zugleich Blattknoten ist, werden die Bestelleinträge auch im Blattknoten eingetragen.

2. Das Bestelldatum wird entweder aus dem aktuellen Knoten ermittelt oder, falls im aktuellen Knoten nichts angegeben wurde, aus dem Wurzelknoten entnommen. Analog gilt dies für die Lieferadresse, für den Lieferzuschlag und für den Paketdienst.

   ```
   Bestellung
   aktDatum = if Datum->isEmpty
              then wurzel.Datum
              else Datum
              endif
   ```

3. Die aktuelle Lieferanschrift.

```
Bestellung
aktLieferadresse = if Lieferanschrift->isEmpty
                   then wurzel.Lieferanschrift
                   else Lieferanschrift
                   endif
```

4. Der aktuelle Lieferzuschlag.

```
Bestellung
aktLieferzuschlag = if Lieferzuschlag->isEmpty
                    then wurzel.Lieferzuschlag
                    else Lieferzuschlag
                    endif
```

5. Der aktuelle Paketdienst.

```
Bestellung
aktPaketdienst = if Paketdienst->isEmpty
                 then wurzel.Paketdienst
                 else Paketdienst
                 endif
```

6. Bestelleinträge können nur bei einem Blattknoten zurückgegeben werden, da beim Anlegen von Teilbestellungen die Bestelleinträge immer in die Teilbestellungen wandern. Ansonsten wird eine leere Menge zurückgegeben.

```
Bestellung
aktBestelleinträge = if teilbestellung->isEmpty
                     then Bestelleintrag
                     else Set
                     endif
```

Die Änderung der Vielfachheit von einer Bestellung zum Bestelleintrag muß auch hier analog zum Bottom–Design geändert werden (siehe Seite 43).

4.2.5 Eintrag in einem Pfad (Path–Ansatz)

Der dritte Ansatz, der im Rahmen dieser Arbeit als letzter betrachtet wird, befaßt sich mit der Datenablage innerhalb eines Pfades. Von einem Blattknoten bis zum Wurzelknoten sind alle Attribute mindestens einmal besetzt. Wurde das selbe Attribut mehrmals mit verschiedenen Werten besetzt, so gilt derjenige Wert, der dem Blatt am nächsten ist, denn der Pfad wächst nach unten und somit werden dort die aktuelleren Werte eingetragen.

Es gelten auch hier wieder die allgemeinen Invarianten aus dem Teilabschnitt 4.2.2 auf Seite 37. Gesamtpreis, Gesamtgewicht, Status und Kunde werden auch wieder wie in den vorausgehenden Teilabschnitten in jedem Knoten besetzt. Die Bestelleinträge werden bei jeder Teilung immer in die Blattknoten verschoben.

Ergänzend zu den allgemeinen Invarianten müssen für diesen Ansatz noch folgende hinzugefügt werden.

1. Zu jedem Blatt muß sich in der Ahnenreihe[7] eine Wertzuweisung für die Attribute Datum, Lieferadresse, Lieferzuschlag, Paketdienst und Bestelleintrag befinden.

   ```
   Bestellung
   blätter->forAll(
     ahnen->select(Datum->notEmpty)->size > 0 and
     ahnen->select(Lieferadresse->notEmpty)->size > 0 and
     ahnen->select(Lieferzuschlag->notEmpty)->size > 0 and
     ahnen->select(Paketdienst->notEmpty)->size > 0 and
     ahnen->select(Bestelleintrag->notEmpty)->size > 0)
   ```

2. Bei der Ermittlung des aktuellen Datums wird der Pfad ausgehend vom Blattknoten durchsucht, bis ein Eintrag im Attribut Datum gefunden wird. Nach der 1. Invariante muß es einen solchen Eintrag geben.

   ```
   Bestellung
   aktDatum = if Datum->isEmpty
              then vater.aktDatum
              else Datum
              endif
   ```

3. Die Lieferadresse wird analog zum Datum berechnet.

[7] Definition der Ahnen im Teilabschnitt 4.2.2 auf Seite 37: „Allgemeine Invarianten“.

```
Bestellung
aktLieferadresse = if Lieferadresse->isEmpty
                   then vater.aktLieferadresse
                   else Lieferadresse
                   endif
```

4. Der Lieferzuschlag wird ebenfalls analog zum Datum ermittelt.

```
Bestellung
aktLieferzuschlag = if Lieferzuschlag->isEmpty
                    then vater.aktLieferzuschlag
                    else Lieferzuschlag
                    endif
```

5. Auch der Paketdienst wird analog zum Datum festgestellt.

```
Bestellung
aktPaketdienst = if Paketdienst->isEmpty
                 then vater.aktPaketdienst
                 else Paketdienst
                 endif
```

6. Bestelleinträge sind nur einem Blattknoten zugeordnet. Aus diesem Grund können sie nur bei Blattknoten geliefert werden. Bei anderen Knoten wird eine leere Menge zurückgegeben.

```
Bestellung
aktBestelleinträge = if teilbestellung->isEmpty
                     then Bestelleintrag
                     else Set
                     endif;
```

Die Änderung der Vielfachheit von einer Bestellung zum Bestelleintrag wird auch in diesem Ansatz nötig, da Nicht–Blattknoten keine Bestelleinträge zugeordnet sind.

4.2.6 Ergebnis

Die Anwendung der drei vorgestellten Ansätze, „Eintrag in jedem Blatt (Bottom–Ansatz)“ im Teilabschnitt 4.2.3, „Eintrag in der Wurzel (Top–Ansatz)“ (4.2.4) und „Eintrag in einem Pfad“ (4.2.5) unterscheidet sich durch die redundante Informationsspeicherung und durch den Informationszugriff.

Beim Bottom–Design wird der Zugriff auf die Information optimiert. Sollen die Bestelldaten zu einer Teilbestellung ermittelt werden, muß nur ein einzelner Blattknoten betrachtet werden, aus dem alle Informationen extrahiert werden können. Dies bedeutet aber eine hohe Redundanz. Das Bestelldatum bleibt meistens innerhalb einer Bestellung konstant und muß dennoch für jede einzelne Teilbestellung abgelegt werden. Auch bei der Lieferadresse und beim Lieferzuschlag sowie beim Paketdienst stellt sich das Problem der redundanten Speicherung. Wird jedoch eine redundante Speicherung akzeptiert und ein schneller Zugriff gewünscht, so können hier die Daten zu einer Teilbestellung in der Ordnung $\mathcal{O}(1)$ ermittelt werden. Zur Realisierung des Bottom–Ansatzes ist die ständige Überprüfung von neun Invarianten in bezug auf die Bestellungsverwaltung nötig. Sechs allgemeine und drei für diesen Ansatz speziell entwickelte Invarianten müssen eingehalten werden.

Der Top–Ansatz reduziert die Redundanz bereits erheblich. Da im Wurzelknoten alle Attribute gesetzt sind und in den Knoten der Teilbestellungen nur die Änderungen gesetzt werden, befinden sich im Baum wesentlich mehr freie Attribute. Nachteilig erweist sich dabei aber, daß beim Auftreten einer Änderung alle nachfolgenden Attribute diese ebenfalls zu setzen haben, um ebenfalls mit konstantem Aufwand die Daten einer Teilbestellung lesen zu können. Für diesen Ansatz sind neben den sechs allgemeinen auch noch weitere sechs spezielle Invarianten nötig, so daß sich insgesamt zwölf Invarianten für den zweiten Ansatz ergeben.

Ebenfalls zwölf Invarianten sind für den Pfad–Ansatz notwendig, d.h. sechs allgemeine und sechs spezielle wie im Top–Ansatz. Da hier ein geändertes Attribut nur einmal zusätzlich zum allgemeinen Wert gespeichert werden muß, ist die Redundanz der gesetzten Attribute in der vorgeschlagenen rekursiven Baumstruktur minimal. Diese Optimierung bezüglich der Speicherung hat den Nachteil, daß maximal alle k Knoten eines Pfades durchsucht werden müssen, was zu einer Ordnung von $\mathcal{O}(k)$ führt.

Ansatz	Bottom	Top	Pfad
Anzahl der Invarianten	6+3	6+6	6+6
Redundanz	hoch	mittel	minimal
Lesender Zugriff	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(k)$

Tabelle 4.1: Ergebnisse aus der Diskussion der verschiedenen Ansätze

In der Tabelle 4.1 sind die Ergebnisse nochmals zusammengefaßt. Aus den Qualitätsanforderungen im Pflichtenheft (siehe Tabelle 3.1 auf Seite 16) ergibt sich, daß von dem zu erstellenden System insgesamt eine „gute“ Effizienz gefordert wird. Diese wäre vom Bottom– und Top–Ansatz, insbesondere dann gewährleistet, wenn man davon ausgeht, daß lesende Zugriffe weitaus häufiger sind als modifizierende Zugriffe. Da aber in der Bestellungsverwaltung auch die alten Bestellungen für Statistikberechnungen verwaltet werden, darf die redundante Speicherung nicht unberücksichtigt gelassen werden. Als Kompromiß kann hier der Top–Ansatz gewählt werden, denn er besitzt ein relativ gutes Zugriffsverhalten und eine mittelmäßige Redundanz. Die Invariantenanzahl wäre zwar beim Bottom–Ansatz noch wesentlich günstiger, aber die hohe Informationsredundanz kann in diesem Knoten bezüglich des hohen Bestelldatenaufkommens nicht akzeptiert werden.

4.3 Vereinigung zum Klassendiagramm der Fachklassen

Abbildung 4.24 auf der nächsten Seite zeigt das Fachklassendiagramm zum Internet–Bookshop nach der Vereinigung der partiellen Fachklassendiagramme.

Die partiellen Klassendiagramme, die in den Teilabschnitten 4.1.1 bis 4.1.6 erarbeitet wurden, werden ohne Attribute und Operationen zu der Gesamtstruktur des Fachklassendiagramms aufgenommen. Die Assoziationen werden, soweit sie bis zu diesem Entwicklungsschritt bekannt sind, in diesem Diagramm eingezeichnet. Neben den Assoziationen mit ihren Vielfachheiten wurden auch die Vererbungsbeziehungen übernommen.

Wie im Klassendiagramm zu sehen ist, ist vor allem die Fachklasse Anschrift mit vielen Klassen assoziiert Die Vertriebe, Paketdienste, Privatkunden und Kontaktpersonen besitzen eine Anschrift. Die Bestellungen und Firmenkunden assoziieren mit dem Referenzattribut „lieferadresse“ ebenfalls auf eine Anschrift.

Die Fachklasse Kontaktperson wird auch aus zwei unabhängigen partiellen Fachklassendiagrammen assoziiert. Sowohl das Kunden–Fachklassendiagramm als auch das Diagramm zur Bestellung stehen in Beziehung mit der Klasse Kontaktperson.

Ein Buch wird vom Bestelleintrag, vom Einkaufspreis, vom Korbeintrag und vom Sachbereich assoziiert. Hierbei wurden die partiellen Diagramme zum Einkaufskorb, zum Buchkatalog und zur Bestellung vereinigt.

Auch die Kundenklasse wurde dreifach verwendet. Im Kundendiagramm, im Einkaufskorb und im Fachklassendiagramm zu den Bestellungen wird es assoziiert.

Als letzter Verknüpfungspunkt ist noch die Bankverbindung zu nennen. Hier wurden die Fachklassendiagramme zu den Kunden und zur Rechnung/Bestellung miteinander verknüpft.

In den partiellen Klassendiagrammen wurde bereits ein „Refactoring“ durchgeführt. Denkbar wäre auch in diesem Stadium, d.h. nach der Zusammenführung der Teildiagramme, eine Anwendung von Verfeinerungen. Da jedoch die Teil–Fachklassendiagramme, abgesehen von den oben genannten Verknüpfungspunkten, untereinander unabhängige Daten modellieren, ergaben sich keine Konflikte bzw. Inkonsistenzen, die bei der Komposition aufzulösen gewesen wären.

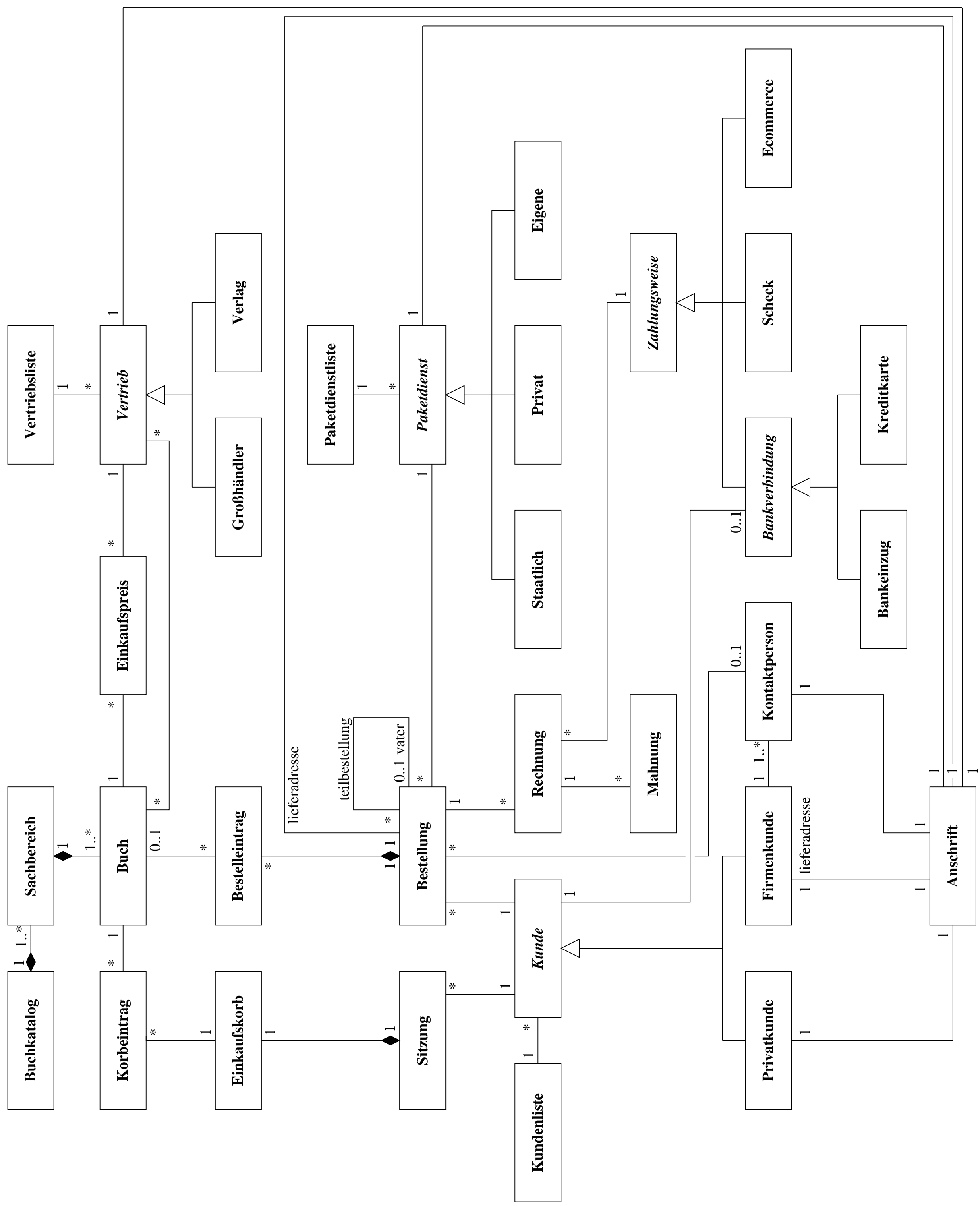


Abbildung 4.24: Gesamt–Fachklassendiagramm zum Internet–Bookshop

Bei der durchgeführten Komposition lag deshalb immer folgender Sachverhalt vor: Falls die Klasse A in einem partiellen Klassendiagramm eine Assoziation zur Klasse B (A–B) und in einem anderen Klassendiagramm eine Assoziation zur Klasse C (A–C) enthält, so existieren im komponierten Fachklassendiagrammdiagramm zwei Assoziationen zur Klasse A (B–A–C).

Andere Probleme, z.B. bezüglich einer Änderung der Vielfachheiten bei der Zusammenführung, traten nicht auf.

4.4 Zusammenfassung

Aus den Produktdaten im Pflichtenheft (siehe Seite 14) wurden in diesem Abschnitt zuerst die partiellen Fachklassendiagramme entwickelt und diese später zu einem Gesamt–Fachklassendiagramm des Internet–Bookshops vereinigt. Bei der Entwicklung der partiellen Diagramme wurde die Struktur verfeinert, bezüglich bestmöglicher Redundanzfreiheit optimiert und nach den Bedürfnissen transformiert. So wurden oftmals gemeinsame Superklassen ermittelt und Wert–Attribute, wenn nötig, zu Referenz–Attributen transformiert. Soweit es möglich und sinnvoll war, wurde ein Refactoring angewendet, wie es in [Opd92] und [OJ93] beschrieben ist. Hierbei kamen vor allem Refactoringtechniken zum Verschieben von Attributen innerhalb der Klassenhierarchie und die Bildung von abstrakten Superklassen zum Einsatz.

In Kapitel 8 auf Seite 135 bei der Betrachtung der Klassenstruktur muß eine nochmalige Anwendung von Refactoring–Techniken analysiert werden. Das Fachklassendiagramm beschäftigt sich mehr mit den Attributen und implizit mit den reinen Zugriffsmethoden. Ein Refactoring bezüglich der funktionalen Bestandteile einer Klasse konnte deshalb hier noch nicht angewandt werden. Z.B. behandelt das angewandte „Refactoring to Generalize: Creating an Abstract Superclass“ neben einer Optimierung bezüglich der Attribute verstärkt auch eine Optimierung der Methoden. Bei einem Fachklassendiagramm in diesem Stadium kann nur eine Optimierung der Attribute stattfinden, da die dynamischen Methoden noch nicht bekannt sind.

Eine Entwicklung des Fachklassendiagramms rein aus den Daten des Pflichtenheftes ohne Bezug auf irgendwelche Funktionsabläufe erlaubt keinen vollständigen Ausblick auf künftige Gemeinsamkeiten und Abhängigkeiten zwischen den Klassen. So wurde das Fachklassendiagramm zumindest teilweise parallel zu den Systemszenarien entwickelt, siehe hierzu Kapitel 5: „Identifikation der Nutzungsfälle“. Beispielsweise müssen bei der Entwicklung des partiellen Fachklassendiagramms zur Bestellung bereits die Abläufe einer Kundenbestellung in Verbindung mit der Rechnungs– und Mahnungserstellung in groben Zügen beschrieben sein, damit später die partiellen Fachklassendiagramme zu einem Gesamt–Klassendiagramm zusammengefügt werden können. Kleinere Unstimmigkeiten könnten zwar beim Zusammenfügen angeglichen werden, jedoch müssen dann ganze Partialdiagramme vollständig überarbeitet und umstrukturiert werden. Hier wäre auch die Anwendung von geeigneten Werkzeugen sinnvoll, die z.B. beim Zusammenfügen gemeinsam benutzte Klassen identifizieren und so z.B. Assoziationen und Erweiterungen selbständig durchführen bzw. vorschlagen. Auch zum Zeichnen wären geeignete Zeichenwerkzeuge sehr hilfreich, die z.B. ein oftmals sehr hilfreiches Vergrößern und Verkleinern erlauben.

Das erstellte Fachklassendiagramm enthält noch keine Klassen zur Benutzerverwaltung. Im Rahmen dieser Fallstudie wird diese nicht weiter betrachtet.

Kapitel 5

Identifikation der Nutzungsfälle

Zur Umsetzung des informellen Pflichtenheftes in einen objektorientierten Entwurf werden in diesem Kapitel die Szenarien auf der Systemebene ermittelt. Aus diesen Systemszenarien ergeben sich die Szenarien auf der Ebene der Nutzungsfälle, die im Kapitel 6: „Interaktionsorientierte Spezifikation“ genauer betrachtet, analysiert und beschrieben werden.

Der Begriff Szenario wird in diesem Zusammenhang synonym zum Nutzungsfall verwendet. Abhängig von den involvierten Objekten unterscheiden sich diese Nutzungsfälle. In diesem Kapitel werden gemäß des Begriffs der Systemszenarien die Nutzungsfälle auf einer Ebene behandelt, bei denen der Nutzer[1] nur mit dem System, betrachtet als ein Objekt, interagiert. Eine weitere Aufteilung der Systemobjekte findet zu diesem Zeitpunkt noch nicht statt.

Im Abschnitt 5.1 wird eine Einführung zu den Szenarien auf Systemebene gegeben, bevor die Kunden–Sitzung (Abschnitt 5.2) und die Händler–Sitzung (Abschnitt 5.3) genauer beschrieben werden. Da sich die Händler–Sitzung bedeutend umfangreicher als die Kunden–Sitzung erweist, wird der Aufgabenkomplex, dem das Händlerpersonal gegenübersteht, in weitere Teilabschnitte untergliedert (Teilabschnitte 5.3.1 bis 5.3.10). In der Zusammenfassung, Abschnitt 5.4, werden vor allem die Probleme erläutert, die sich bei der Erstellung der Systemszenarien ergaben.

5.1 Einführung – Szenarien auf der Ebene des Systems

Im Pflichtenheft wurden bereits die Produktfunktionen erläutert (siehe Abschnitt 3.3 auf Seite 12). Daneben werden bei der gleichzeitig stattfindenden Entwicklung des Anwendungskerns (vgl. Kapitel 4 auf Seite 17) einige Vorgänge noch weiter geklärt. Aus diesen

[1] Der Nutzer wird oftmals auch als Akteur bezeichnet.

beiden Informationsquellen werden in diesem Kapitel nun die Nutzungfälle auf Systemebene herausgearbeitet. Nach [Bre98a] beschreiben Szenarien auf der Systemebene typische Anwendersitzungen mit Folgen aktivierter Nutzungsfälle. Das System wird dabei als Ganzes (black box) betrachtet. Die so erstellten Nutzungsfälle werden dann im Kapitel 6 wieder aufgegriffen, um daraus gemeinsam mit der im Kapitel 4 entwickelten Fachklassenstruktur die Szenarien auf Nutzungsfallebene herauszuarbeiten. Dabei ist es dann möglich, das in den Systemszenarien eingeführte Systemobjekt in seine Bestandteile aufzuteilen (in Objekte des Anwendungskerns und weitere für den Ablauf notwendige Objekte) und die Nachrichten zwischen diesen Klassen zu ermitteln. Der Dialog, der hier auf Systemebene zwischen Nutzer und Systemobjekt stattfindet, wird im nächsten Kapitel ebenfalls genauer spezifiziert.

Nutzungsfälle sind Dienste, die das System dem Anwender anbietet. Genauer: Ein Nutzungsfall ist eine zusammenhängende Interaktion zwischen dem Anwender und dem System, die der Erledigung einer Aufgabe dient (vgl. [Bre98a]).

Zur Darstellung von Nutzungsfällen werden Sequenzdiagramme verwendet. Grundsätzlich sind Sequenzdiagramme Bestandteil der UML 1.3. Da wir in diesem Kapitel jedoch einige Erweiterungen von Sequenzdiagrammen verwenden, werden diese hier vorab erläutert. Die Darstellung orientiert sich an den Beschreibungstechniken aus [BBH+99] und [Bre98a]. Für einen Vergleich verschiedener ähnlicher Beschreibungstechniken für Komponenteninteraktionen verweisen wir auf [Krü99]. Die erweiterten Sequenzdiagramme beinhalten folgende Ergänzungen:

Bezeichner: Sequenzdiagramme in der hier verwendeten Form besitzen einen Bezeichner, so daß sie aus anderen Sequenzdiagrammen referenziert werden können. Der Name erscheint in der linken oberen Ecke des Diagramms.

Referenz: Die Erweiterung beinhaltet einen Referenz–Mechanismus. Das graphische Symbol für eine Referenz ist ein Rechteck, das sich horizontal über alle Lebenslienien im verwendeten Sequenzdiagramm erstreckt. Der Bezeichner im Rechteck kennzeichnet das referenzierte Sequenzdiagramm. Semantisch bedeutet eine solche Referenz, daß das referenzierte Sequenzdiagramm an Stelle des Referenzsymbols substituiert werden kann.

Alternativauswahl: Damit wir in einem Sequenzdiagramm eine Menge von alternativen Abläufen spezifizieren können, ist es möglich, im Rechteck zur Referenzierung auch eine Menge von Bezeichnern anzugeben. Die Bezeichner werden hier durch senkrechte Striche (|) verbunden; anlog dem Operationssymbol "oder" in vielen Programmiersprachen. Semantisch bedeutet diese Alternativauswahl, daß beim Erreichen einer solchen Auswahl diese durch eine der referenzierten Alternativen ersetzt werden kann.

Wiederholung: Wiederholungen werden in den Diagrammen durch eine senkrechte vertikale Linie dargestellt, die oben und unten durch eine kleine horizontale Linie beschränkt wird. Die Länge dieses "Wiederholungs–Indikators" kennzeichnet den zu wiederholenden Abschnitt. Der Bezeichner neben diesem Indikator spezifizert die Anzahl der möglichen Wiederholungen.

Erläuterungen zu Sequenzdiagrammen sind z.B. in [BRJ98], [BRS97], [Bur97], [FS98], [JBR98], [Oes97] und [Rat97] zu finden.

Der Dialog mit dem Benutzer spielt für Nutzungsfälle eine wichtige Rolle. Der Benutzer wird dabei als Akteur bezeichnet. Beim Akteur eines Nutzungsfalls ist nicht die Person an sich entscheidend, sondern die Rolle, die sie in dem Nutzungfall spielt. Personen können dabei auch mehrere Rollen spielen. Z.B. kann ein Mitarbeiter des Internet–Buchhandels sowohl die Rolle des Kunden als auch die Rolle des Händler–Personals spielen.

Für eine in diesem Kapitel folgende Ermittlung und Spezifikation der Systemszenarien findet zuerst eine Ermittlung der Akteure statt. Zu den verschiedenen Akteuren können dann später die Nutzungsfälle betrachtet werden.

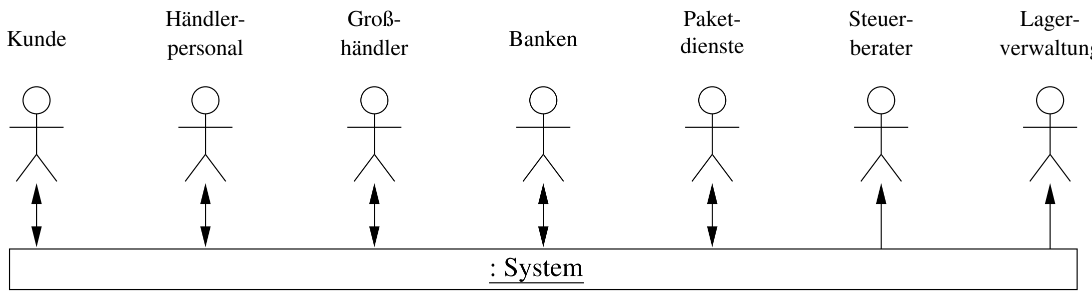


Abbildung 5.1: Akteure im Internet–Bookshop

Abbildung 5.1 zeigt den Internet–Bookshop mit all seinen Akteuren. Die Kunden und das Händlerpersonal führen eine reale Anwendersitzung aus, indem sie beim Arbeiten mit dem System Daten ein– und ausgeben. Die Paketdienste, Großhändler und Banken tauschen ebenfalls mit dem System Daten aus, obwohl hier nur zwei Systeme miteinander kooperieren. Für die Zusammenarbeit mit diesen entfernten Systemen müssen im Bookshop–System neben den Datenausgangs–Funktionen auch Klassen für den Empfang von Nachrichten und Daten bereitgestellt werden, wie dies z.B. auch für eine Kunden–Sitzung notwendig ist.

Zwei Ausnahmen bilden die Verbindungen mit dem Steuerberater und mit der Lagerverwaltung. Da hier Daten nur nach außen fließen und somit keine Daten vom System entgegengenommen werden müssen, bedarf es später auch keiner hierfür bereitgestellten Datenobjekte im Bookshop–System. Diese Verbindungen stellen also nur eine Informationsweiterleitung an einen außenstehenden Akteur dar[2]. Deshalb werden diese beiden Akteure nicht explizit für die Ermittlung von Nutzungsfällen herangezogen. Die hierfür nötige Analyse und

[2]In Abbildung 5.1 wurde diese Unterscheidung durch eine andere Darstellung der Pfeile verdeutlicht. Die Pfeile führen hier nur vom System weg.

das Design werden vor allem bei der Ermittlung der Szenarien zur Händler–Sitzung und teilweise zur Kunden–Sitzung übernommen.

Somit ergeben sich aus der Betrachtung der Akteure zum Internet–Bookshop folgende fünf Bereiche, zu denen Nutzungsfälle erstellt werden können. Die Aktionen des Kunden, des Händlerpersonals, der Großhändler, der Banken und der Paketdienste mit dem System können genauer betrachtet werden. Bei dieser Betrachtung können die Produktfunktionen herangezogen werden und bei der Bearbeitung des Systemszenarios je nach zugehörigem Akteur verwendet werden.

Ausgehend von den Produktfunktionen wurde hier vorab zur Ermittlung der Systemszenarien eine Analyse der verschiedenen Akteure durchgeführt. Sie dient in erster Linie zu einer Einteilung der Systemszenarien.

In den folgenden Abschnitten 5.2 und 5.3 werden die Systemszenarien der Kunden– und Händler–Sitzung eingehend analysiert. Eine eigene Betrachtung der Großhändler–, Banken– und Paketdienst–Sitzung findet im Rahmen dieser Arbeit nicht statt, es werden nur die mit der Händlersitzung relevanten Teile im Abschnitt Händler–Sitzung betrachtet. Ein vollständiges Ausarbeiten aller Sitzungen würde den Rahmen dieser Arbeit sprengen.

5.2 Kunden–Sitzung

Abbildung 5.2 auf der nächsten Seite beschreibt die Systemszenarien zur Kunden–Sitzung, welche dem Bookshop–Nutzer eine beliebige Folge von Suchanfragen, Bestellungen und Verfassen von Lesermeinungen ermöglichen.

Wie oben beschrieben werden die für die Kunden–Sitzung relevanten Produktfunktionen auf dem Pflichtenheft hier für die Erstellung der Systemszenarien verwendet. Die Bedeutung der Systemszenarien und Referenzen auf die speziellen Produktfunktionen werden im folgenden beschrieben.

Diesen Aktionen geht nur ein Anwendungsstart voraus. Eine Authentifizierung wird nur für eine Bestellung oder für das Verfassen einer Lesermeinung benötigt. Aus diesem Grund erfolgt diese beim konkreten Bearbeiten einer dieser Funktionen, entfällt aber bei reinen Suchanfragen, in denen sich z.B. ein Kunde nur mal im voraus über geeignete Bücher informiert. Die Authentifizierung muß deshalb nicht bei jedem Anwendungsstart erfolgen. Ein explizites Abmelden erfolgt ebenfalls nicht, so daß das Programm einfach beendet und verlassen wird (Teilbild 5.2 a).

Eine Suche kann entweder durch Eingeben von Suchkriterien oder durch sequentielles Blättern im Buchkatalog erfolgen (Teilbilder 5.2 b und c). Durch eine Selektion mittels Suchkriterien kann eine Liste mit keinem, einem oder mehreren Büchern ausgegeben werden. Diese Suchfunktion entspricht der im Pflichtenheft erläuterten Produktfunktion */F:Suche/* auf Seite 12.

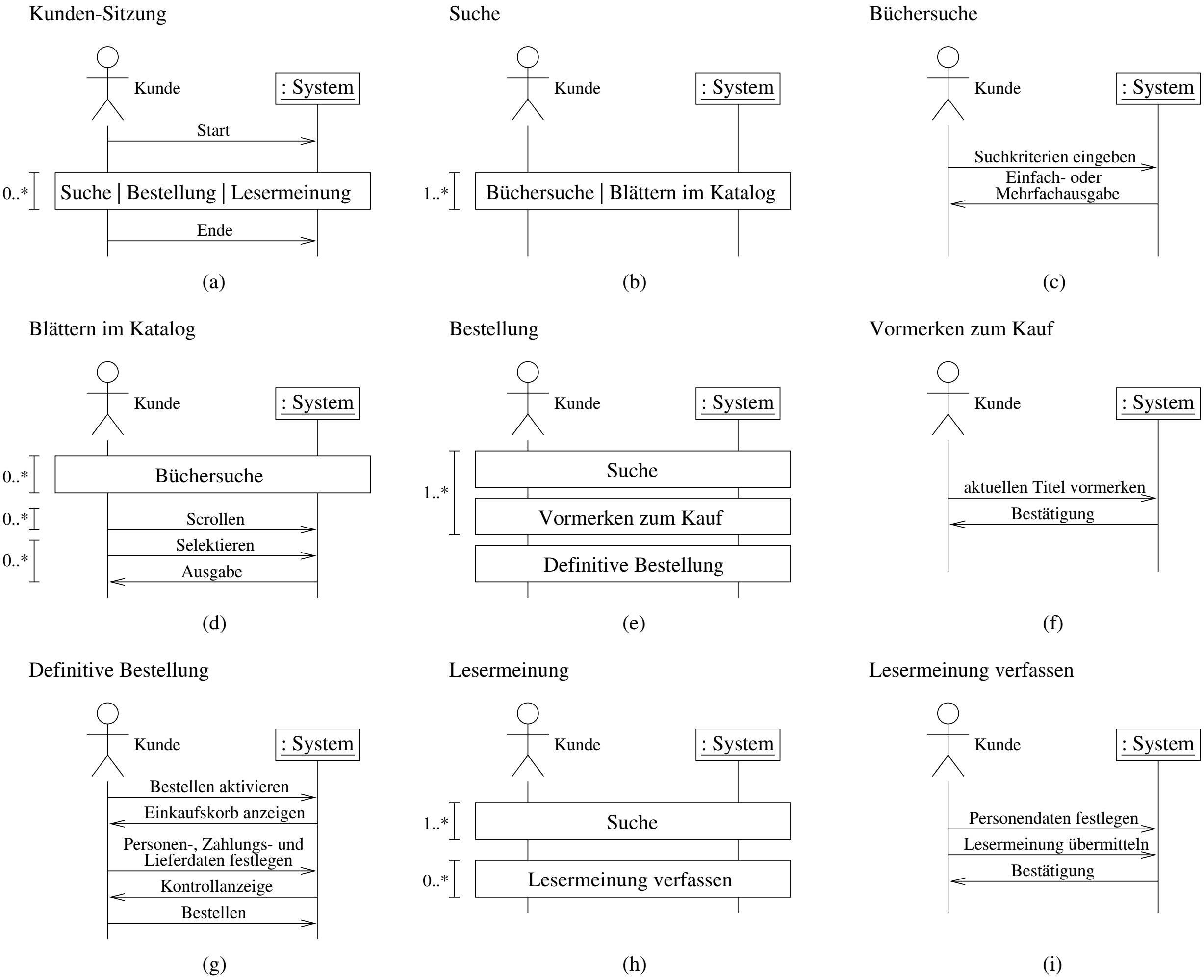


Abbildung 5.2: Kunden–Sitzung

Da der Gesamtbestand im Bookshop–Katalog sehr groß sein kann und aus diesem Grund ein sequentielles Durchblättern durch alle Einträge oftmals nicht sinnvoll wäre, kann dem Blättern eine Suche vorgeschaltet werden. Die Ergebnismenge kann durch wiederholtes Ändern der Suchanfrage auf eine geeignete Größe angepaßt werden (Query by Example). Wie in Teilbild 5.2 d zu sehen ist, kann entweder durch den Gesamtbestand oder durch einen Auszug gescrollt werden und zu den Titeln eine genauere Information angefordert werden. Dieses Szenario leitet sich aus der Pflichtenheftfunktion */F:Blättern/* auf Seite 13 ab.

Eine Bestellung kann erst erfolgen, wenn im Einkaufskorb die gewünschten Bücher enthalten sind (Pflichtenheftfunktion */F:Vormerken/* auf Seite 13). Nachdem ein Buch entweder sequentiell oder durch eine konkrete Buchsuche gefunden wurde, kann es in den Einkaufskorb aufgenommen werden, so daß es zum Kauf vorgemerkt wird (Teilabbildungen 5.2 e, f). Nachdem alle gewünschten Bücher durch einen solchen Vorgang selektiert wurden, kann zur definitiven Bestellung übergegangen werden.

Vor dem Bestellen wird der Einkaufskorb zur Kontrolle nochmals angezeigt. Falls die Einkaufsliste korrekt ist, müssen noch die Personendaten, Zahlungsweise und Lieferdaten in ein entsprechendes Formular eingetragen werden. Daraufhin werden die gesamten Bestelldaten dem Käufer nochmals präsentiert, und mit einer positiven Quittung wird dann der Bestellvorgang abgeschlossen (Abbildung 5.2 g). Dieser System–Nutzungsfall leitet sich aus der Pflichtenheftfunktion */F:Bestellung/* auf Seite 13 ab.

Eine letzte Funktion, die dem Bookshop–Nutzer vom System zur Verfügung gestellt wird, stellt das Verfassen von Lesermeinungen dar (Pflichtenheftfunktion */F:Lesermeinung/* auf Seite 13). Wie im Teilbild 5.2 h zu sehen ist, muß hierbei zuerst das Buch, zu dem eine Kritik oder Empfehlung abgegeben werden soll, gesucht werden.

Vor dem Schreiben bzw. Übermitteln des Eintrags müssen aber noch die Personendaten angegeben werden, da diese teilweise gemeinsam mit der Lesermeinung auf der Buchseite angezeigt werden. Nach dem Eintreffen einer Lesermeinung im System gibt dieses an den Verfasser noch eine Bestätigung aus (Teilbild 5.2 i).

Die Systemszenarien Büchersuche, Blättern im Katalog, Vormerken zum Kauf, Definitive Bestellung und Lesermeinung verfassen konnten direkt aus aus Produktfunktionen abgeleitet werden (es wurde dabei eine entsprechende Referenz angegeben). Die restlichen hier erzeugten Nutzungsfälle beschreiben vor allem Vorgangsabläufe, bei denen die bereits genannten Nutzungsfälle verwendet werden. Auch sie lassen sich aus den Pflichtenheftfunktionen ableiten. Jedoch müssen hier oftmals Informationen aus mehreren Funktionsbeschreibungen kombiniert werden.

5.3 Händler–Sitzung

Die Händler–Sitzung befaßt sich mit dem gesamten Aufgabenkomplex, dem das Händler–Personal gegenübersteht. Sie befaßt sich mit Verwaltungsaufgaben, die durch Buchbestellungen entstehen, ermöglicht aber auch Bank– und Steuerberater–Aktionen sowie die Programmverwaltung selbst. Abbildung 5.4 zeigt die inhärenten Aufgaben der Händler–Sitzung.

Auch zur Händler–Sitzung werden wie im vorausgehenden Kapitel wieder die Nutzungsfälle aus den Produktfunktionen ermittelt. Falls zusätzlich Information aus dem Fachklassendiagramm verwendet wird, wird dies explizit angegeben.

In Abbildung 5.3 ist das Sequenzdiagramm zum Nutzungsfall Händler–Sitzung angegeben. In Abhängigkeit vom User–Modus, der vom System nach der Eingabe des Benutzernamens und des Kennworts ermittelt wird, müssen gegenüber der Anwendung von bestimmten Programmfunktionen Beschränkungen eingeführt werden. Die Programmverwaltung darf z.B.

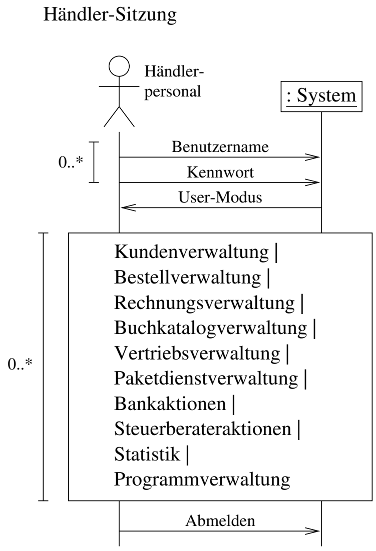


Abbildung 5.3: Händler–Sitzung

nicht von jedem Personalmitglied ausgeführt werden, da hier z.B. Kennwörter überschrieben werden können[3]. Aus diesem Grund wird der Nutzungsfall Händler–Sitzung weiter verfeinert. Es erfolgt eine Teilung bezüglich der ausführbaren Funktionen.

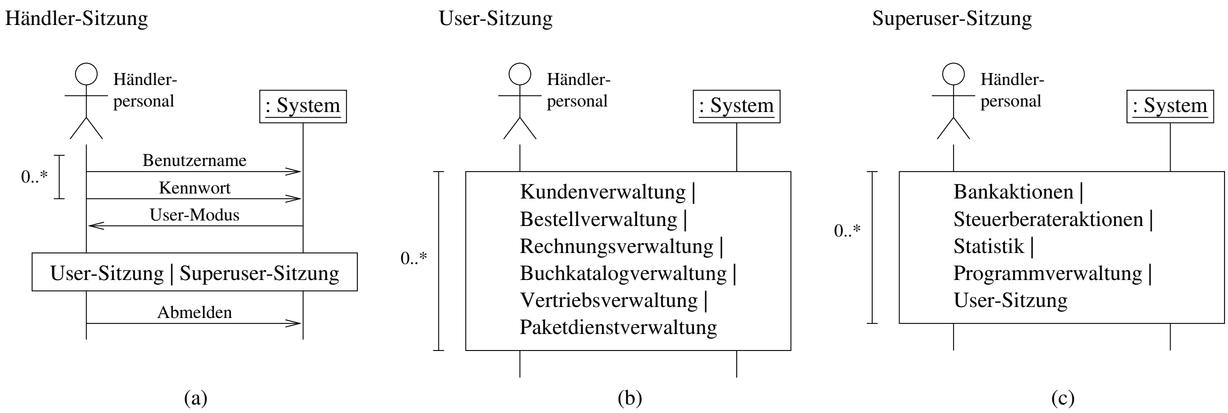


Abbildung 5.4: Verfeinerte Händler–Sitzung

Abbildung 5.4 zeigt das Sequenzdiagramm nach der Teilung. Je nach Anwender (normaler Anwender, engl. User, oder Administrator, engl. Superuser) stehen verschiedene Funktionen zur Verfügung.

Hierbei ist noch anzumerken, daß die User–Sitzung sowohl aus dem Stellvertreter von der

[3]Die Funktion der Programmverwaltung wurde im Pflichtenheft nicht explizit beschrieben. Sie muß deshalb hier beim Softwareentwurf ergänzt werden.

Händler–Sitzung wie auch aus dem Stellvertreter aus der Superuser–Sitzung referenziert wird. Wenn die User–Sitzung aus der Superuser–Sitzung entfernt wird, kann er über den Stellvertreter in der Händler–Sitzung auch noch alle Aufgaben erledigen. Jedoch wurde die User–Sitzung hier in die Superuser–Sitzung mitaufgenommen, da sie das direkte Anspringen der User–Sitzung erlaubt. Es wurde also hier in das Diagramm weitere Information, die über eine normale Referenzierung hinausgeht, mit aufgenommen, denn bereits hier werden die späteren Ablaufvorgänge bereits festgelegt. Im folgenden erfolgt wieder eine Beschreibung des Sequenzdiagramms.

Im Gegensatz zur Kunden–Sitzung verlangt die Händler–Sitzung eine Benutzeranmeldung. Wie üblich werden hierzu Benutzerkennwort und Paßwort eingegeben, worauf das System anschließend den Benutzermodus ermittelt. Im User–Modus stehen dem Personal die Funktionen zum Bearbeiten von Bestellvorgängen inklusive Verwaltungstätigkeiten bereit (Abbildung 5.4 b). Der Superuser–Modus (Abbildung 5.4 auf der vorherigen Seite c) erlaubt zusätzliche Aufgaben wie Bankgeschäfte, Steuerberateranbindung und Systemverwaltung. Auch Statistikoperationen sind nur einem vertrauten Anwenderkreis zugänglich, da hier z.B. unternehmensrelevante Veränderungen ersichtlich sind.

Die Systemszenarien zu den einzelnen Aufgaben des Händler–Personals werden in den folgenden Teilabschnitten 5.3.1 bis 5.3.9 ermittelt und dargestellt. Teilabschnitt 5.3.10 liefert noch eine Verbesserungsmöglichkeit.

5.3.1 Kundenverwaltung

Das Szenario der Kundenverwaltung wird von der Produktfunktion */F:Kundenverwaltung/* im Pflichtenheft auf Seite 13 abgeleitet. Sie befaßt sich mit der Modifikation von Kundendatensätzen im System. Die gewöhnlichen Operationen sind hierzu das Anlegen, Ändern und Löschen eines Kunden (siehe Abbildung 5.5 a). Da jedoch bei Kunden zwischen Privatkunden und Firmenkunden unterschieden werden muß, hat das System die hierzu notwendigen Funktionen ebenfalls zu unterstützen.

Bei der Erfassung des partiellen Fachklassendiagramms zu den Kundendaten (siehe Teilabschnitt 4.1.1 auf Seite 18) wurde bereits entschieden, daß zu einem Firmenkunden immer mindestens eine Kontaktperson bestehen muß. Deshalb muß das System hier bei der Kundenverwaltung auch die Verwaltung von Kontaktpersonen miteinbeziehen.

Bei der Operation „Kunden anlegen“ ergibt sich somit eine Vierteilung. Neben Privat– und Firmenkunden müssen auch Kontaktpersonen angelegt werden können und darüberhinaus existiert auch noch die wahrscheinlich am häufigsten benutzte Funktion zum Übernehmen von Kundendaten aus Bestellungen (Operation „Bestellkunden übertragen“). Das zugehörige Sequenzdiagramm ist im Teilbild 5.5 b zu sehen.

Die trivialste Funktion hieraus bildet das Anlegen von Privatkunden (Teilbild 5.5 d). Es werden nacheinander die Daten eines Kunden manuell eingegeben. Nach einer Kontrollan-

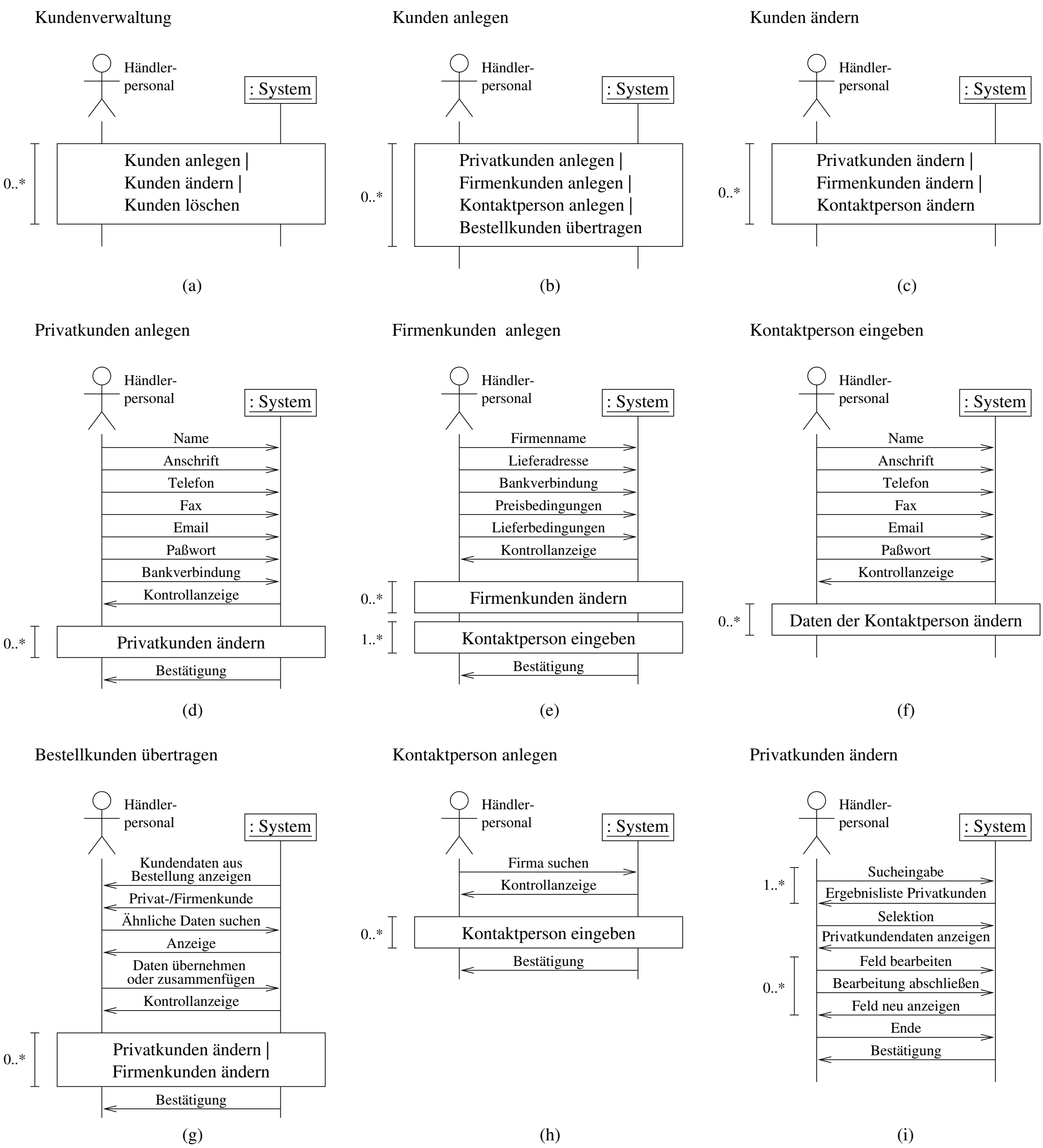


Abbildung 5.5: Kundenverwaltung als Funktion der Händler–Sitzung (Teil 1)

zeige des Systems besteht noch die Möglichkeit zum Ändern von eventuell falsch erfaßten Daten. Zum Abschluß bestätigt das System noch die Aufnahme des Datensatzes.

Handelt es sich um einen Firmenkunden (Teilbild 5.5 e), werden vorerst nur die firmen-

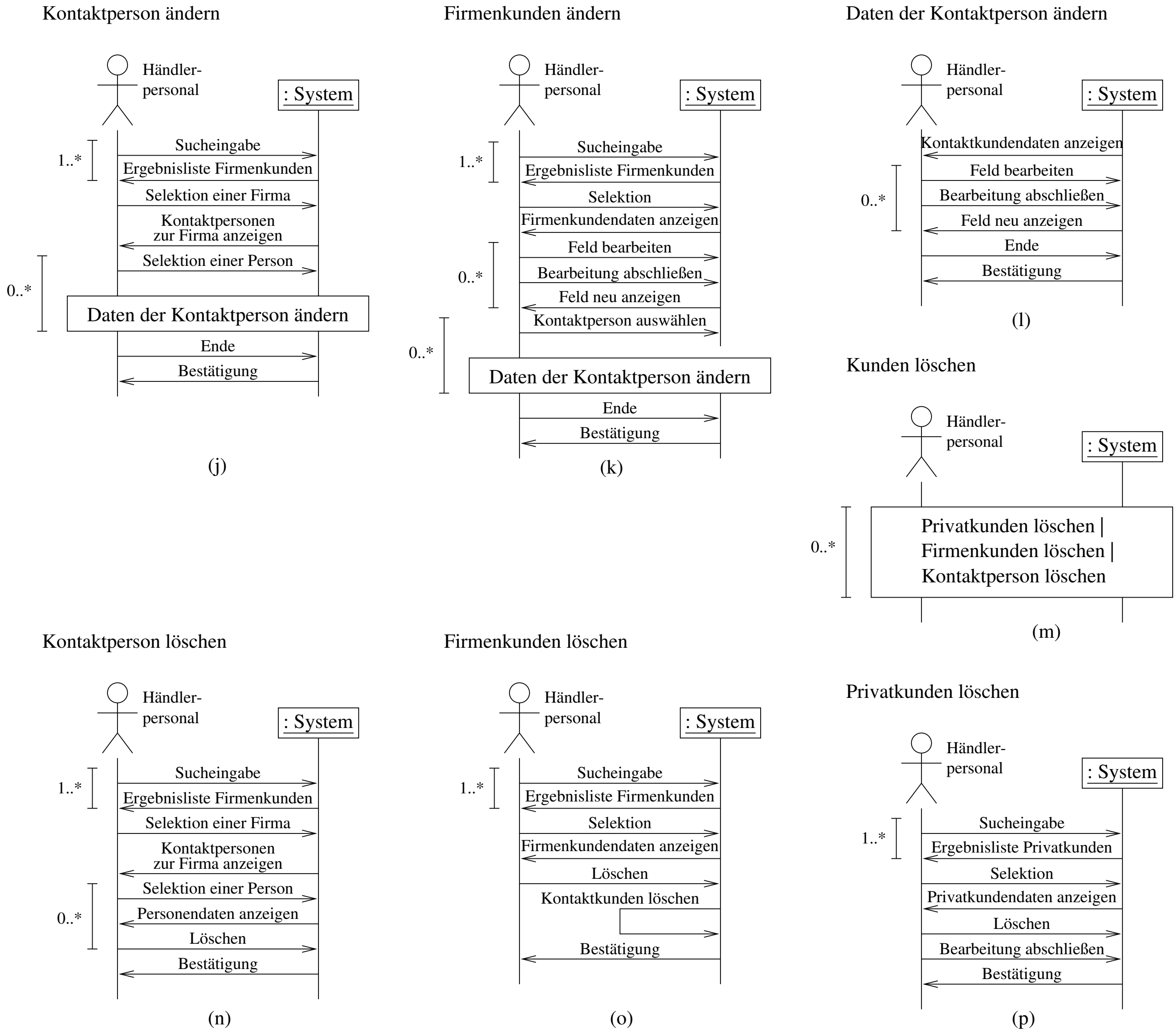


Abbildung 5.6: Kundenverwaltung als Funktion der Händler–Sitzung (Teil 2)

spezifischen Daten manuell eingegeben und, falls notwendig, geändert. Eine oder mehrere Kontaktpersonen, die mit dem Internet–Bookshop in Verbindung treten, werden im Anschluß separat angegeben.

Die Daten einer Kontaktperson werden entweder beim Anlegen einer Firma eingegeben oder können auch separat hinzugefügt werden. Dieser zweite Fall ergibt sich dann, wenn z.B. zu einer Firma eine neue Kontaktperson hinzugefügt werden soll. Hierbei muß zuerst die Firma, zu der die neue Person hinzugefügt werden soll, im System gesucht (Teilbild 5.5 h) werden und daraufhin können die Kontaktpersonendaten eingegeben werden (Teilbild 5.5 f).

Bei Bestellungsaufgabe werden im Fall einer Erstbestellung die Kundendaten angegeben. Diese werden in der Funktion „Bestellkunden übertragen“ (Teilbild 5.5 g) visuell dargestellt. Zusätzlich gibt das System aus, ob es sich hierbei um einen Privat– oder Firmenkunden handelt. Anschließend werden ähnliche Datensätze im System ermittelt, so daß z.B. bei einer falsch geschriebenen Anschrift nicht ein Neukunde angelegt wird, sondern daß die Möglichkeit einer Verbesserung besteht. Die Daten können daraufhin bei einem Neukunden übernommen werden oder z.B. bei einer neuen Kontaktperson zu einem Firmendatensatz hinzugefügt werden. Nach einer Kontrollanzeige besteht nochmals eine Änderungsmöglichkeit. Nach Abschluß der Funktion erfolgt vom System noch eine Bestätigung über die Datensatzaufnahme.

Neben den verschiedenen Einfüge–Operationen existieren auch die Modifikations– und Lösch–Operationen für Privat– und Firmenkunden, sowie für Kontaktpersonen (Teilbilder 5.5 c und 5.6 m).

Sowohl beim Löschen (Teilbild 5.6 p) wie auch beim Ändern (Teilbild 5.5 i) eines Privatkunden muß zuerst der Datensatz im System ermittelt werden. Im ersten Fall, d.h. wenn der Datensatz aus dem System entfernt werden soll, muß der Anwender nur noch das Löschen des speziellen Datensatzes bestätigen. Beim Ändern können die Daten einzeln mehrfach modifiziert werden, bis der Anwender die Bearbeitung beendet. Sowohl nach dem Löschen wie auch nach dem Ändern gibt das System bei erfolgreichem Operationsabschluß eine Bestätigung aus.

Auch beim Ändern eines Firmenkunden (Teilbild 5.6 k) muß der Datensatz hierzu erst ermittelt werden. Nach einer Modifikation der firmenspezifischen Daten können auch noch die Daten der Kontaktpersonen abgeändert werden (Teilbild 5.6 l). Die Operationen „Firmenkunden ändern“ und „Daten der Kontaktperson ändern“ werden bei Erfolg wieder vom System bestätigt.

Soll hingegen nur eine Kontaktperson geändert werden und die Firmendaten gleich bleiben, so müssen nicht erst die Firmendaten ermittelt werden. Es kann sofort der Datensatz der Kontaktperson gesucht und die Datenmodifikation anschließend durchgeführt werden (Teilbild 5.6 j).

Das Löschen einer Kontaktperson ist analog möglich. Es wird der Datensatz gesucht und das Löschen bestätigt (Teilbild 5.6 n). Wird ein Firmenkunde gelöscht (Teilbild 5.6 o), erfolgt nach dem Suchen und Löschen der Firmendaten auch die Entfernung aller Kontaktpersonen.

Anmerkung zu den Abbildungen 5.5 und 5.6: Die Operationen „Privatkunden anlegen“, „Firmenkunden anlegen“ und „Bestellkunden übertragen“ verweisen auf die Operationen „Privatkunde ändern“ und „Firmenkunden ändern“. Bei diesen beiden Änderungsfunktionen findet zu Beginn eine Kundenauswahl statt, die bei den Funktionen zum Kundenanlegen übergangen werden müßte. Alternativ dazu könnte auch in „Kontaktperson ändern“

und „Firmenkunden ändern“ die Kundenauswahl als optional markiert werden. Durch eine Kontextbedingung müßten wir dann fordern, daß die Kundenauswahl höchstens dann fehlen darf, wenn die jeweilige Änderungsoperation im Kontext der entsprechenden Anlegeoperation auftritt. Für eine ausführliche Diagrammdarstellung müßten noch zwei weitere Sequenzdiagramme für die Operationen „Daten der Privatkunden ändern“ und „Daten der Firmenkunden ändern“ eingefügt werden. Da dies jedoch die Anzahl der Sequenzdiagramme noch weiter erhöhen würde, wurden sie weggelassen. Im Fall der Kontaktperson wurde eine Funktion „Daten der Kontaktperson ändern“ exemplarisch eingefügt.

5.3.2 Bestellverwaltung

Einerseits müssen vom System die Bestellungen der Kunden verwaltet werden, jedoch muß auch eine Vertriebsbestellung möglich sein, bei der sich der Internet–Bookshop die zur Auslieferung nötigen Bücher besorgt. Diese beiden Aufgaben, sowie eine Lager–Kontaktierung sind die Arbeitsbereiche der Bestellverwaltung als Funktion der Händler–Sitzung. Eine Zusammenfassung der Teilfunktionen zeigt Abbildung 5.7 a. Ein Abschluß mit einer Bestätigung findet jeweils am Ende einer Operation statt. Die Pflichtenheftfunktion */F:Bestellungsverwaltung/* auf Seite 13 beschreibt diese Operation bereits verbal.

Die Verwaltung von Kundenbestellungen und Bestellungen des Bookshops bei Vertriebspartnern findet beide Male in den Bestellklassen aus dem Fachklassendiagramm in Abbildung 4.9 auf Seite 25 statt. Die Unterscheidung zwischen Kunden– und Vertriebsbestellung liefert die zugehörige Kunden–Assoziation. Bei Bestellungen vom Bookshop wird ein reservierter Kunde eingetragen, wodurch keine zusätzliche Klassenstruktur zur Verwaltung von Vertriebsbestellungen nötig wird. Außerdem können hierdurch die Manipulationsoperationen zum Hinzufügen, Löschen und Ändern von Bestellungen und Bestelleinträgen sowohl auf die Kunden– wie auch für die Vertriebsbestellungen angewendet werden.

Wird manuell eine Bestellung hinzugefügt, d.h. nicht automatisch wie beim Übernehmen des Einkaufskorbs bei der Kundenbestellung, müssen Kundendaten, Lieferadresse, –zuschlag und Paketdienst angegeben werden. Das aktuelle Datum und der Bestellstatus „neu“ werden vom System ergänzt (Teilbild 5.7 b). Zu der neu erstellten Bestellung können sofort Bestelleinträge hinzugefügt, gelöscht und geändert werden.

Zum Löschen einer Bestellung (siehe auch im Pflichtenheft: */F:Stornierung/* auf Seite 13) muß zuerst das Datenobjekt der Bestellung ermittelt werden, bevor alle Bestelleinträge und die Bestellung selbst gelöscht werden. Eine Bestätigung signalisiert dem Benutzer den gewünschten Erfolg der Operation (Teilbild 5.7 c). Beim Ändern einer Bestellung erfolgt nach der Datenobjekt–Ermittlung die Modifikation der Bestelldaten mit anschließender Manipulation der Bestelleinträge (Teilbild 5.7 d).

Die Operationen „Bestelleintrag hinzufügen“ (Teilbild 5.7 e) und „Bestelleintrag ändern“ (Teilbild 5.7 g) werden nur von den Operationen „Bestellung hinzufügen“ und „Bestel-

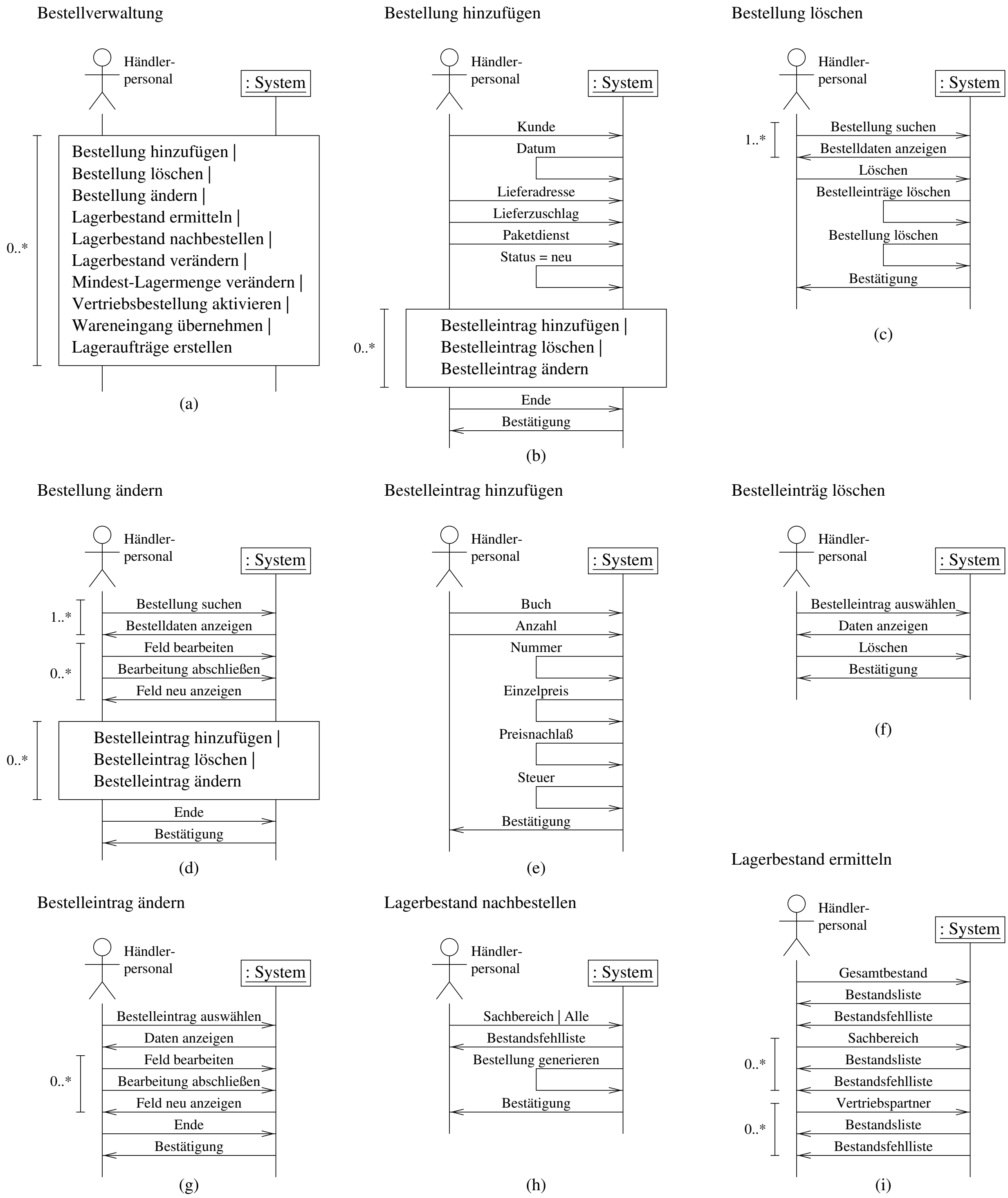


Abbildung 5.7: Bestellverwaltung als Funktion der Händler–Sitzung (Teil 1)

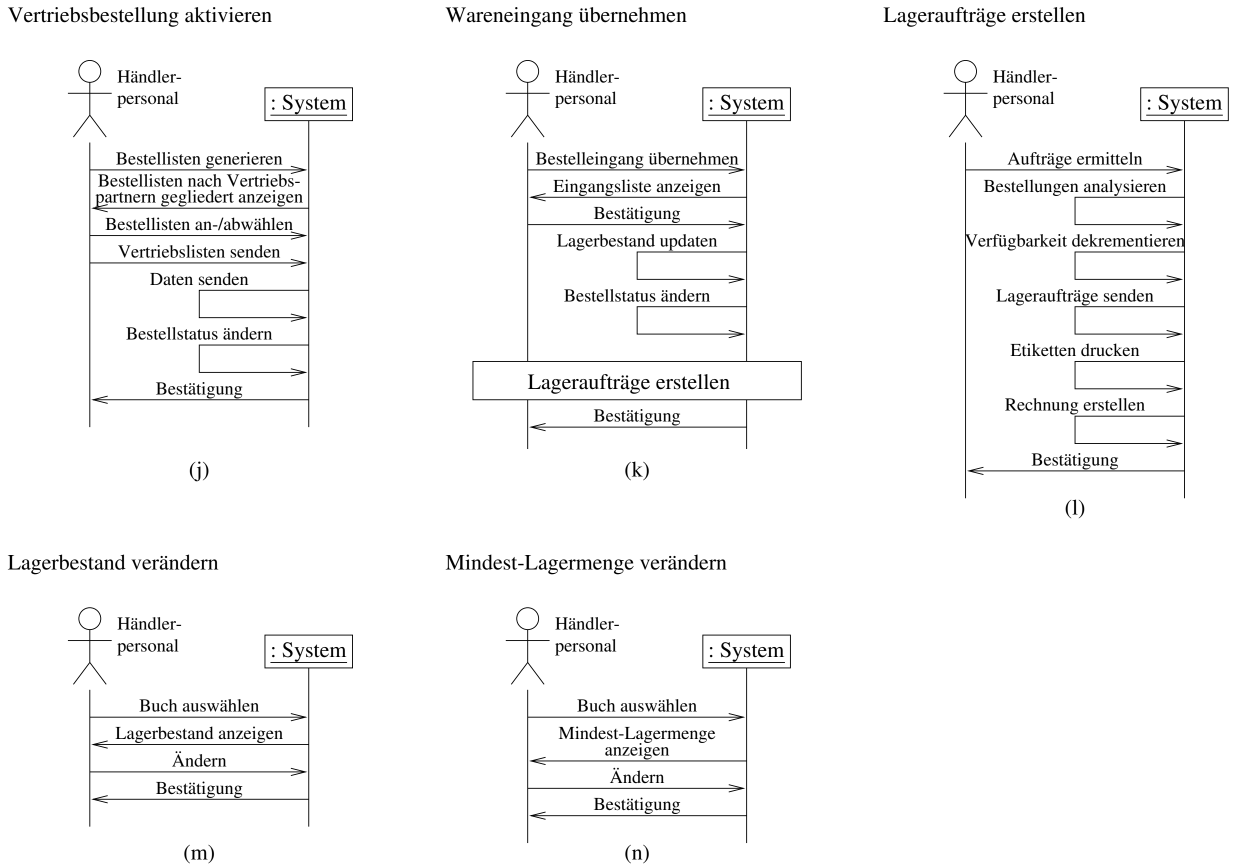


Abbildung 5.8: Bestellverwaltung als Funktion der Händler–Sitzung (Teil 2)

lung ändern“ aufgerufen, so daß bei Operationsbeginn bereits eine zugeordnete Bestellung existiert, d.h. sie muß nicht erst innerhalb der Funktionen ausgewählt werden. Beim Hinzufügen müssen Buch und Anzahl manuell eingegeben werden, während die laufende Nummer, welche für eine Referenz z.B. auf dem Rechnungsbeleg von Bedeutung sein kann, der Einzelpreis, der Preisnachlaß und die zu entrichtende Steuer vom System ermittelt bzw. aus den Buch– und Kundendaten übernommen werden. Beim Ändern muß genauso wie beim Löschen eines Bestelleintrags dieser aus den einer Bestellung zugeordneten Einträgen ausgewählt werden. Die Felder können nach und nach abgeändert werden. Das Löschen eines Bestelleintragobjekts muß nach einer Kontrollanzeige quittiert werden, um die Löschung durchzuführen (Teilbild 5.7 f).

Dem Händlerpersonal obliegt auch die Lagerbestandsverwaltung. So muß in gewissen Zeitabständen auch die Lagerbestandsliste kontrolliert werden. Der Bearbeiter kann sich informieren, welche Bücher in absehbarer Zeit nachbestellt werden müssen. Wie Teilbild 5.7 i zeigt, wird erst der Gesamtbestand ausgegeben und eine Liste der Bücher, deren Lagerbestand zu klein ist. Diese Liste wird als Bestandsfehlliste bezeichnet.

Anschließend können die Bestandsliste und die Bestandsfehlliste zu einem Sachbereich oder zu einem Vertriebspartner aufgelistet werden. Die Auflistung nach den Vertriebspartnern erscheint zu dem Zweck sinnvoll, daß, wenn bezüglich eines Vertriebspartners ein besonders großer Fehlbestand vorliegt, eine außerplanmäßige Bestellung bei diesem eingeleitet wird.

Zum Ermitteln einer Bestandsfehlliste muß eine Mindest–Lagermenge zu jedem Buch definiert sein. So kann z.B. von besonders häufig gekauften Büchern ein großer Lagerbestand gewünscht werden, während z.B. seltene Fachbücher oder veraltete Literatur gar nicht im Lager vorrätig sein sollen. Da kein Attribut in der Fachklasse Buch für eine Aufnahme einer Mindest–Lagermenge vorgesehen ist, muß die Fachklasse um dieses ergänzt werden (siehe Abbildung 5.9). Bei Büchern, die im Lager aus obigen Gründen nicht vorrätig gehalten werden sollen, wird die Mindest–Lagermenge auf Null gesetzt.

Abbildung 5.9: Ergänzung der Fachklasse Buch

Wird nun von einem Kunden eine Bestellung eines Buches aufgegeben, so wird der Lagerbestand dekrementiert. War der Lagerbestand bereits bei Null, so wandert der Lagerbestand auch in den negativen Bereich.

Soll nun der Lagerbestand wieder aufgefüllt werden, so müssen zu den Vertriebspartnern Bestellungen gesendet werden. Diese Bestellungen werden nach dem Anstoßen der Funktion „Lagerbestand nachbestellen“ (Teilbild 5.7 h) durch das Händlerpersonal zu einem oder mehreren Sachbereichen generiert. Hierzu wird über die Bücher iteriert und die Differenz aus Mindest–Lagermenge und Lagerbestand gebildet. Ist das Ergebnis größer Null, wird ein Bestelleintrag für eine Vertriebsbestellung generiert.

Die so erstellten Bestellungen liegen nun analog zu den Kundenbestellungen im Einkaufskorb vor. Die Vertriebsbestellungen müssen nun noch aktiviert werden (Teilbild 5.8 j). Dabei werden aus den neuen Vertriebsbestellungen Bestellisten für die verschiedenen Vertriebe (Großhändler und Verlage) generiert. Der Anwender hat noch die Möglichkeit, Bestellungen für bestimmte Vertriebe abzuwählen, und die weiterhin aktivierten Bestellisten werden anschließend z.B. mit Email an die Vertriebe weitergeleitet. Der Status in den Bestellungen wird bei erfolgreicher Bestellungs–Zustellung von „neu“ auf „offen“ geändert.

Im Fall einer Warenanlieferung von einem Vertriebspartner werden die Waren im Lager auf Vollständigkeit kontrolliert. Falls hierbei keine Mängel auftreten, startet ein Sachbearbeiter aus dem Händlerpersonal die Operation „Wareneingang übernehmen“ (Teilbild 5.8 k). Die Wareneingangsdaten, die z.B. vom Vertriebspartner ebenfalls mittels Email übermittelt worden sind, werden in die Bestelldaten eingebracht. Dabei werden die Lagerbestände und der Bestellstatus aktualisiert.

Daraufhin können die nun zur Erfüllung möglichen Lageraufträge an das Lager übermittelt werden (Operation „Lageraufträge erstellen“ in Teilbild 5.8 l). Dabei werden zuerst die noch ausstehenden Kunden–Bestellungen analysiert. Die möglichen Bestellungen werden nun durchgeführt, wobei die Verfügbarkeit dekrementiert, ein Lagerauftrag an das Lager gesendet, ein Etikettendruck und abschließend eine Rechnungserstellung gestartet wird. Zum Etikettendruck wurde die Pflichtenheftfunktion */F:Etikettendruck/* auf Seite 14 als verbale Funktionsbeschreibung verwendet.

Weiterhin hat das Händler–Personal auch die Möglichkeit, den Lagerbestand eines Buches (Teilbild 5.8 m) und die Mindest–Lagermenge zu verändern (Teilbild 5.8 n).

5.3.3 Rechnungsverwaltung

Während einerseits die Bestellungen und dadurch auch die Lieferungen vom Händlerpersonal verwaltet werden müssen, spielt andererseits die Rechnungsverwaltung ebenso eine entscheidende Rolle. Sie erlaubt dem Personal ein manuelles Erstellen und Manipulieren von Rechnungen und Mahnungen (siehe Abbildung 5.10 a auf Seite 69). Die automatische Rechnungserstellung erfolgt, wie im Abschnitt 5.3.2 auf Seite 64 beschrieben, zusammen mit dem Senden der Bücher in der Operation „Lageraufträge erstellen“ (Abbildung 5.8 auf Seite 66, Teilbild l). Ausgangspunkt für die Rechnungsverwaltung ist die Pflichtenheftfunktion */F:Rechnungserstellung/* auf Seite 14.

Wurde z.B. durch das Fehlen von Kundenangaben eine Rechnungserstellung bei der Auslieferung nicht in die Wege geleitet, muß diese Funktion manuell gestartet werden. Wie Teilbild 5.10 b zeigt, wird hierzu die zugehörige Bestellung ausgewählt und die Rechnungserstellung gestartet. Das System erstellt daraufhin automatisch ein Rechnungsobjekt und füllt dieses mit den zugehörigen Daten im System. Die Vorgänge Drucken, Kuvertieren und Frankieren erfolgen ebenfalls automatisch.

Diese Funktion darf aber nicht mit dem „Rechnung hinzufügen“ verwechselt werden. In diesem Fall werden zu einer ausgewählten Bestellung die Rechnungsdaten von Hand eingegeben (Teilbild 5.10 c). Erforderlich wird diese Funktion z.B. dann, wenn ein Kunde nicht den vollen Rechnungsbetrag begleicht und deshalb eine zusätzliche Rechnung zur Anforderung des Restbetrags ausgestellt werden muß. Ein Modifizieren des Standardrechnungstexts kann vor dem Drucken, Kuvertieren und Frankieren ebenfalls noch erfolgen.

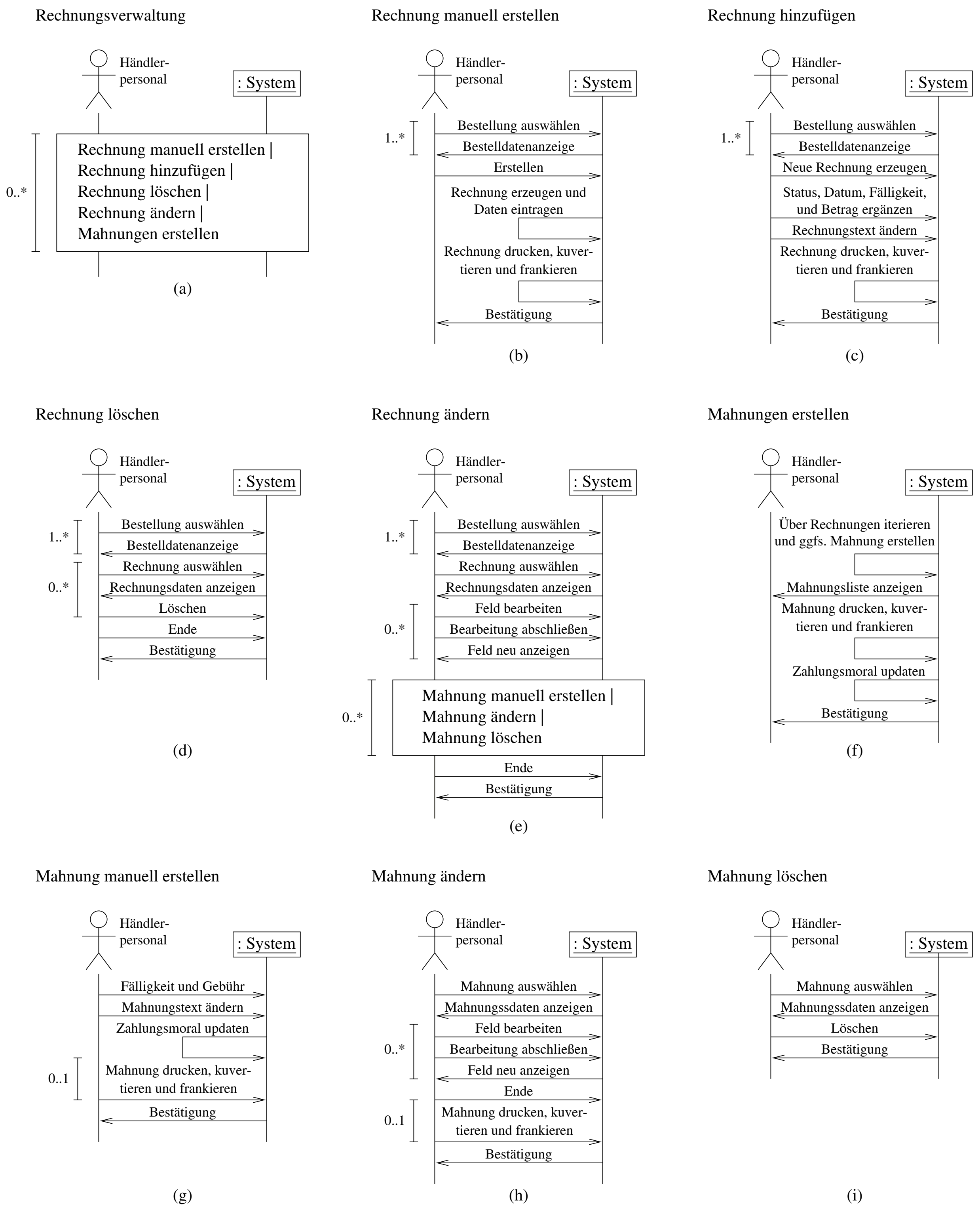


Abbildung 5.10: Rechnungsverwaltung als Teil der Händler–Sitzung

Kann z.B. eine Auslieferung eines Buchartikels nicht zur Erfüllung kommen, da beispielsweise ein Buch beim Versand beschädigt wird, muß auch die Rechnung gelöscht werden. Teilbild 5.10 d zeigt dieses Systemszenario.

Beim Ändern einer Rechnung findet nach der Bestellungsauswahl eine Möglichkeit zur Modifikation der einzelnen Rechnungseinträge im Rechnungsobjekt statt. So können der Status, das Rechnungsdatum, die Fälligkeit, der Betrag, die Mehrwertsteuer und der Rechnungstext nacheinander verändert werden (Teilbild 5.10 e). Da sich Mahnungen immer auf Rechnungen beziehen, erfolgt auch die gesamte manuelle Mahnungsverwaltung als Teilfunktion der Rechnungsänderung. Dies birgt den Vorteil, daß bei den Operationen „Mahnung manuell erstellen“, „Mahnung ändern“ und „Mahnung löschen“ keine Bestellungsauswahl vorgeschaltet werden muß.

Die automatische Mahnungserstellung muß zwar vom Händlerpersonal angestoßen werden, erfolgt aber ansonsten ohne eine zusätzliche Informationseingabe. Nach einem Feststellen der fälligen und noch nicht bezahlten Rechnungen erfolgt eine Anzeige, die dem Anwender nochmals eine Kontrollübersicht bietet. Wurde z.B. in einer manuell erstellten Bestellung ein falsches Datum eingegeben, so kann eine Mahnungsabsendung noch verhindert und das Datum korrigiert werden. Nach dem Drucken, Kuvertieren und Frankieren erfolgt auch eine Aktualisierung der Zahlungsmoral des Kunden (vgl. Teilbild 5.10 f).

Die Modifikationsoperationen zu den Mahnungen erlauben wieder ein manuelles Erstellen, ein Ändern und ein Löschen (siehe Teilbilder 5.10 g, h, i). Die trivialste Funktion bildet das Löschen, bei dem nach der Mahnungsauswahl diese entfernt wird. Vor dem Aktualisieren der Zahlungsmoral findet beim manuellen Erstellen eine Dateneingabe und eine Modifikation des Mahnungstextes statt. Das Drucken, Kuvertieren und Frankieren kann bereits hier oder erst später durch eine Mahnungsänderung eingeleitet werden. Neben diesen Tätigkeiten zur Versandübergabe erlaubt die Mahnungsänderung natürlich auch eine Datenänderung in einem Mahnungsobjekt.

5.3.4 Buchkatalogverwaltung

Die Manipulation des Buchkatalogs, welcher einen wesentlichen Datenbestand beinhaltet, erfolgt vom Händlerpersonal unter dem Punkt Buchkatalogverwaltung, welche in der Pflichtenheftfunktion */F:Katalogbearbeitung/* auf Seite 14 bereits verbal beschrieben wurde. Die Dreiteilung in Buchkatalog, Sachbereich und Buch, wie sie im Fachklassendiagramm zum Buchkatalog entwickelt wurde (siehe Abschnitt 4.1.4 auf Seite 28), erlaubt eine Verwaltung der Sachbereiche, die ihrerseits eine Verwaltung der Bücher ermöglichen. Hieraus ergibt sich, daß der Buchbestand nur über die Sachbereiche modifiziert werden kann.

Dem Händlerpersonal stehen dafür die Manipulationsoperationen zum Modifizieren des Sachbereichs neben einer Buchdatenübernahme und einer Sachbereichsänderung für Bücher zur Verfügung (Teilbild 5.11 a auf Seite 71).

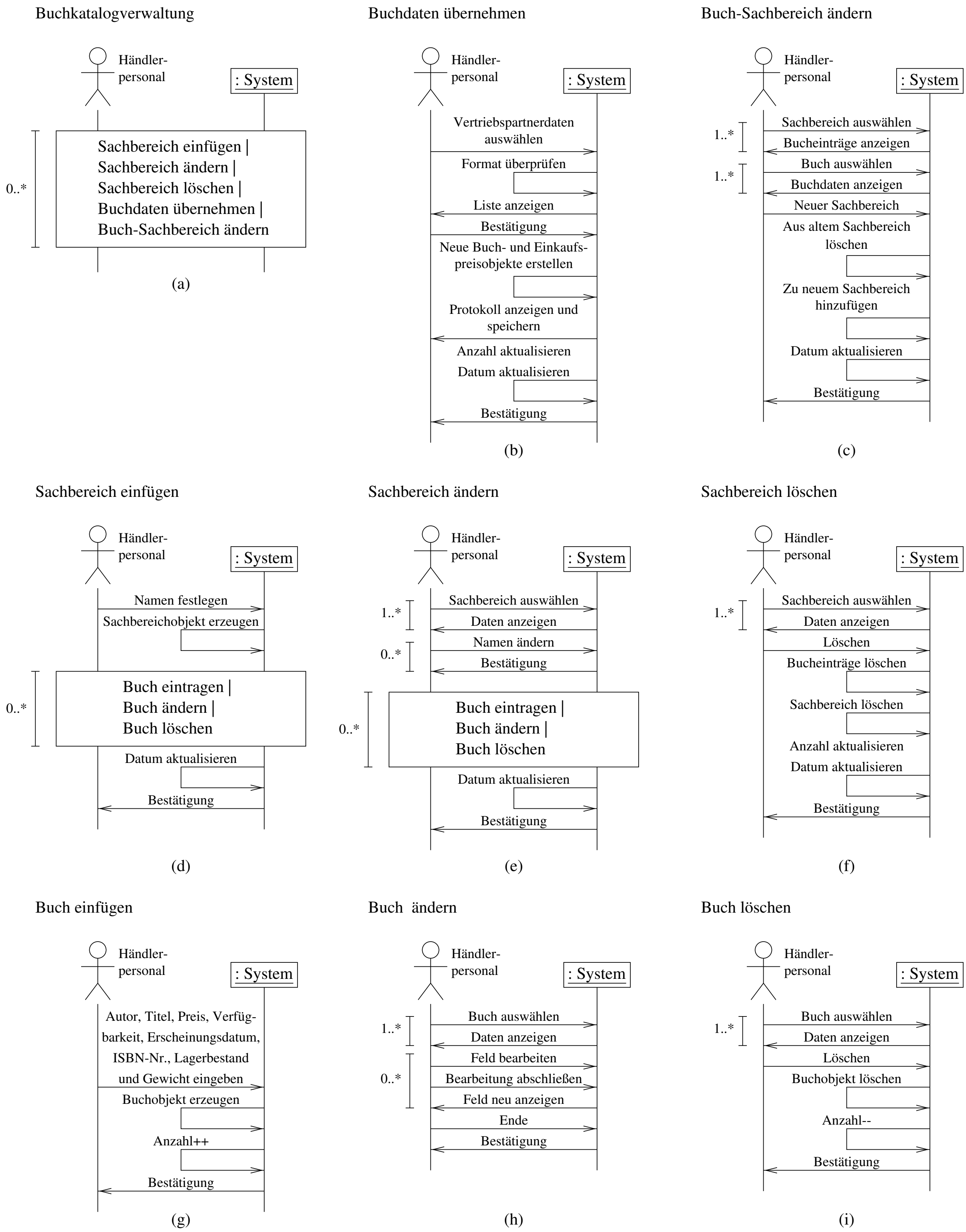


Abbildung 5.11: Buchkatalogverwaltung als Funktionen der Händler–Sitzung

Die Funktion „Buchdaten übernehmen“ (Teilbild 5.11 b) findet Anwendung, falls von einem Vertriebspartner ein Buchkatalog in einem verwertbaren Datenformat vorliegt. Das Personal wählt die einzuspielenden Daten aus, bevor das System nach einer Formatüberprüfung die Daten zur Kontrolle nochmals anzeigt. Wird daraufhin eine Bestätigung zur Übernahme abgegeben, werden zu jedem Buch ein Buch– und ein Einkaufspreisobjekt erstellt. Letzteres spiegelt den Bezugspreis des Buches über den speziellen Vertriebspartner wider, von dem der gerade einzuspielende Buchkatalog vorliegt. Während der Datenübernahme werden Fehler in eine Protokolldatei aufgezeichnet und dem Anwender, in diesem Fall dem Händlerpersonal, präsentiert. Zum Abschluß dieser Operation wird die Buchanzahl der im Katalog existierenden Einträge und das Änderungsdatum aktualisiert.

Soll ein Buch einem anderen Sachbereich zugeordnet werden, z.B. wenn ein neuer Sachbereich ergänzt wird, muß das Buch im alten Sachbereich gesucht, von diesem entfernt und zum neuen hinzugefügt werden (siehe Teilbild 5.11 c). Ein Aktualisieren des Buchkatalog–Änderungsdatums hat ebenfalls stattzufinden.

Eine vollständige Manipulation des Sachbereichs wird wieder durch die üblichen Modifikationsoperationen erreicht. Zum Einfügen eines neuen Sachbereichs wird gemäß Teilabbildung 5.11 d dessen Namen eingegeben, worauf vom System ein neues Objekt erzeugt wird. Nach dem Abschluß von Buchergänzungen und –modifikationen erfolgt noch wie bei allen anderen Änderungen zum Sachbereich eine Aktualisierung des Änderungsdatums, bevor das System die Operation bestätigt. Ein Ändern erfordert eine Sachbereichsauswahl, worauf der Sachbereichsname modifiziert und Bucheinträge geändert werden können (Teilbild 5.11 e). Beim Löschen muß nach der Auswahl dieses bestätigt werden, bevor die zugehörigen Buchobjekte und das Sachbereichsobjekt gelöscht werden (Teilbild 5.11 f).

Die nun noch fehlenden Manipulationsoperationen für die Bücher erlauben eine Neueingabe, eine Modifikation von bestehenden Büchern und ein Löschen. Neben der Dateneingabe beim Hinzufügen und einer Selektion zum Löschen muß hierbei jeweils die Anzahl der Buchkatalogseinträge aktualisiert werden (siehe Teilbilder 5.11 g und i). Beim Ändern können wie gewohnt die einzelnen Felder manipuliert werden (Teilbild 5.11 h).

5.3.5 Vertriebs– und Paketdienstverwaltung

Die Aufgaben zur Vertriebs– und Paketdienstverwaltung behandeln im wesentlichen die Aktualisierung der Vertriebs– und Paketdienstlisten, wie diese im Pflichtenheft unter dem Abschnitt Produktfunktionen (3.3 auf Seite 12) unter den Punkten */F:Großhändlerverwaltung/* und */F:Paketdienstverwaltung/* verbal beschrieben wurden.

Die Großhändlerverwaltung wurde zur Vertriebsverwaltung umbenannt und enthält jetzt zusätzlich die Verlage, die ihrerseits Bücher selbst ausliefern.

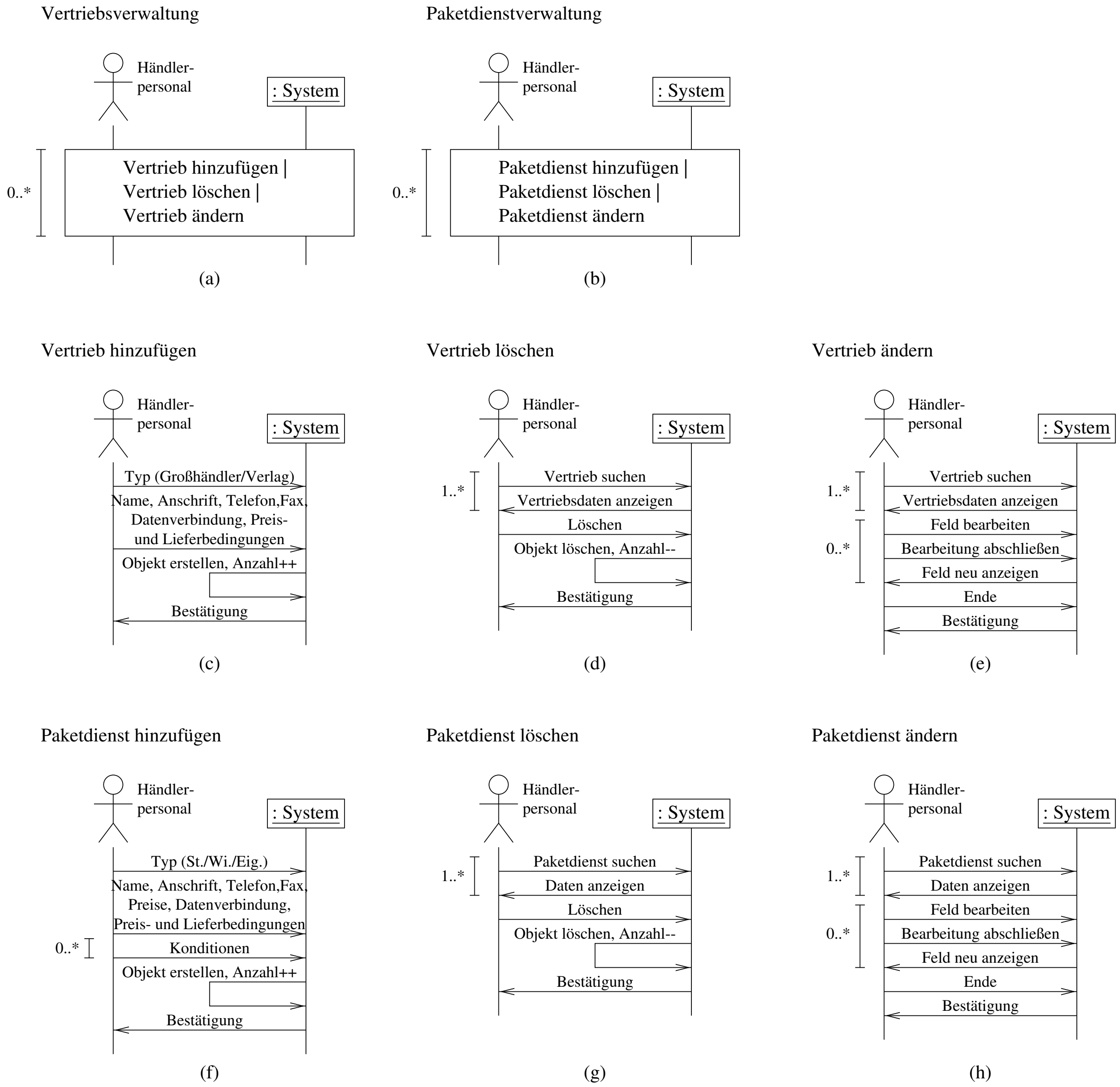


Abbildung 5.12: Vertriebs– und Paketdienstverwaltung als Funktionen der Händler–Sitzung

Zur Aufrechterhaltung der Paketdienste und Vertriebe sind wieder die üblichen Modifikationsoperationen nötig. Es wird jeweils ein Hinzufügen, Ändern und Löschen von Einträgen unterstützt (siehe Teilabbildungen 5.12 a, b).

Beim Eintragen neuer Partner werden die Daten angegeben und neue Listeneinträge erstellt (Teilabbildungen 5.12 c, f). Den Änderungs– und Löschfunktionen folgt nach einer Auswahl entweder die Bearbeitung der einzelnen Felder (Teilabbildungen 5.12 e, h) oder das definitive Löschen (Teilabbildungen 5.12 d, g).

5.3.6 Bankaktionen

In diesem und in den nächsten beiden Abschnitten erfolgt nur noch eine kurze stichpunktartige Beschreibung der Verwaltungsfunktionen des Händlerpersonals. Da die hierzu nötigen Sequenzdiagramme keine besonderen Schwierigkeiten enthalten und somit diese Arbeit nur in die Länge ziehen würden, wird von deren exakter Ausarbeitung abgesehen.

Die Bankaktionen wurden im Pflichtenheft unter dem Punkt */F:Datenaustausch/* auf Seite 14 bereits erwähnt. Folgende Operationen wären denkbar und könnten als exemplarische Sequenzdiagramme auf Systemebene ausgearbeitet werden: Kontobewegungen ermitteln, Überweisungen tätigen, Stornierungen einleiten, Fehlgeschlagene Buchungen abrufen sowie Zinsen und Gebühren einholen.

5.3.7 Steuerberateraktionen

Die Steuerberaterfunktionen wurden ebenfalls unter dem Punkt */F:Datenaustausch/* im Pflichtenheft bereits beschrieben. Als wichtigste Funktion ist hier nur die Übermittlung der Buchungen an den Steuerberater zu nennen. Weitere Aufgaben wurden hier vorerst nicht spezifiziert.

5.3.8 Statistik

Statistikberechnungen sollen Auskunft und Vergleichszahlen über das Unternehmen bereitstellen. Im Pflichtenheft wurde auch hierzu bereits eine Beschreibung unter dem Punkt */F:Statistik/* aufgenommen. Weitere Einzelheiten hierzu werden in bezug auf diese Arbeit ebenfalls weggelassen.

5.3.9 Programmverwaltung

Als ein neuer Aufgabenkomplex, der im Pflichtenheft noch nicht berücksichtigt wurde, kommt die Benutzerverwaltung zum zu erstellenden Software–System hinzu. Beispielsweise sind folgende Operationen zu nennen: Benutzer anlegen, Benutzerrechte festlegen und ändern sowie Datensicherung und Datenrestaurierung.

Auch das Klassendiagramm enthält hierzu noch keine Fachklassen, die diese Verwaltung und Speicherung unterstützen. Da diese Operationen und Klassen jedoch jederzeit ergänzt werden können und vorerst für den weiteren Entwurf nicht unbedingt notwendig sind, werden sie hier nicht näher betrachtet.

5.3.10 Bemerkungen zur Händler–Sitzung

Nachdem die Operationen der Händler–Sitzung vorgestellt wurden, fällt dabei auf, daß viele Funktionen, die vom Anwender manuell aufgerufen werden, in gewissen Zeitabschnitten automatisch ausgeführt werden können. Für das System ist derzeit keine Klasse vorgesehen, die z.B. die Operationen „Lagerbestand nachbestellen", „Vertriebsbestellung aktivieren" und „Lageraufträge erstellen" der Bestellverwaltung oder Statistikoperationen nach einem abgelaufenen Zeitintervall ausführt. Die Händler–Sitzung könnte dahingehend noch erweitert werden.

Bei der Erstellung der Sequenzdiagramme ergab sich bezüglich der Kunden–Sitzung folgender Unterschied. Bei der Kundensitzung wurden die Produktfunktionen im Pflichtenheft so genau spezifiziert, daß zu jeder Pflichtenheftfunktion nur ein oder zwei Sequenzdiagramme entstanden. Bei der Händler–Sitzung hingegen waren so viele Alternativen zu betrachten, daß zu einer einzigen Pflichtenheftfunktion z.B. 16 Teildiagramme (bei der Kundenverwaltung) entwickelt werden mußten. Dies liegt auch daran, daß die Händler–Szenarien viele manuelle Eingriffe in das System erlauben.

Die Verfeinerung eines Nutzungsfalls wurde beim Nutzungsfall Händler–Sitzung gezeigt. Wie zu sehen ist, können Verfeinerungen nicht nur bei der Klassenstruktur angewendet werden, sondern ebenfalls bei Systemszenarien. Verfeinerungen werden auch in den späteren Kapiteln immer wieder auftreten.

5.4 Zusammenfassung

Wie bereits schon in der Zusammenfassung von Kapitel 4: „Die Struktur des Anwendungskerns" erwähnt wurde, kann weder das Fachklassendiagramm noch die Beschreibung der Systemszenarien völlig losgelöst voneinander stattfinden. Die Reihenfolge der Kapitel 4 und 5 wurde willkürlich gewählt, sie spiegelt die Bearbeitungsreihenfolge in diesem Fall nicht wider. Die Bearbeitung dieser beiden Kapitel erfolgte parallel.

Die Darstellung der Systemszenarien mit Hilfe von exemplarischen Sequenzdiagrammen, wie dies in [Bre98a] und [Bre98b] vorgeschlagen und hier verwendet wurde, erwies sich als überaus sinnvoll. Das dynamische Verhalten des Systems wurde ausgehend von der verbalen Beschreibung im Pflichtenheft exemplarisch festgelegt, wobei hieraus auch ein Ableiten der weiteren Varianten vereinfacht wird. Es können zu einem späteren Zeitpunkt Teile der Sequenzdiagramme wiederverwendet und die entscheidenden Teilsequenzen verändert werden. Außerdem müssen hierzu in den bereits existierenden Sequenzdiagrammen weitere Auswahloptionen, sogenannte Stellvertreter (siehe [Bre98a]), eingegliedert werden.

Durch die Anwendung von Sequenzdiagrammen mit Stellvertretern entstehen bei wenigen Optionen bereits eine Vielzahl von Sequenzdiagrammen. Die einzelnen Sequenzdiagramme sind dann oftmals sehr kurz und enthalten z.B. nur eine einzelne Aktion gefolgt von einem weiteren Stellvertreter. Dies führt vor allem zu einer sehr zeitaufwendigen Erstellung und zu sehr auseinandergezogenen und deshalb oftmals unübersichtlichen Diagrammen, besonders dann, wenn zur Darstellung nur die normale Seitengröße eines DIN A4 Blattes zur Verfügung steht, weil z.B. die Diagramme in einen Bericht eingebettet werden müssen. Im Teilabschnitt 5.3.1: „Kundenverwaltung“ wurde hierzu bereits das Problem mit dem Anlegen der Privat– und Firmenkunden erläutert (siehe Seite 63). Bei den Sequenzdiagrammen könnte diesbezüglich, d.h. zur besseren Darstellung von Alternativ– und Schleifenabläufen, noch eine Weiterentwicklung stattfinden.

Eine Alternative zu den Sequenzdiagrammen mit Stellvertretern wäre hier die Verwendung von Highlevel Message Sequence Charts. Diese werden im nächsten Kapitel bei der Identifikation der Nutzungfälle erläutert und verwendet.

Da es sich bei den Sequenzdiagrammen nur um exemplarische Abläufe handelt, werden im Normalfall nicht alle Sonderfälle behandelt. Auch wenn mehrere Varianten des Ablaufs erstellt werden, bleiben weitere Sonderfälle offen. Über das Fehlgehen von Operationen mit den Auswirkungen auf das System, oftmals einem Funktionsabbruch, wird im Diagramm zu diesem Zeitpunkt oft noch nichts ausgesagt.

Über die Genauigkeit der Sequenzdiagramme kann keine generelle Aussage erfolgen. Einige Funktionen behandeln die Ein- und Ausgaben sehr ausführlich, inklusive interner Systemnachrichten (siehe z.B. Teilabbildung 5.12 c auf Seite 73), während bei anderen z.B. ein komplexer Suchvorgang nur durch die Aktion „Suchen“ beschrieben wird. Oftmals wird aus Vereinfachungsgründen eine Kurzform gewählt, obwohl zu diesem Zeitpunkt noch keine Entscheidung über die erforderliche Genauigkeit gefällt werden kann. Die Diagrammerstellung erfordert dann einen Kompromiß, dessen Richtigkeit sich erst in späteren Bearbeitungsschritten herausstellt. Sollte jedoch eine zu ungenaue Beschreibung vorliegen, kann diese zu einem späteren Zeitpunkt ergänzt werden.

Kapitel 6

Interaktionsorientierte Spezifikation

Im vorausgehenden Kapitel wurden die Interaktionen zwischen Nutzer und „System als Ganzes" betrachtet. Außerdem wurden wichtige interne Abläufe auf der abstrakten Ebene des Systems ermittelt. Sie wurden dort in den Sequenzdiagrammen als Nachrichtenschleifen des Systemobjekts eingezeichnet.

In diesem Kapitel werden nun sowohl die Aktionen zwischen Nutzer und System als auch die systeminternen Interaktionen genauer analysiert. Die abstrakte Klasse System wird durch die in Kapitel 4 erzeugten Fachklassen und durch zusätzliche Vorgangsklassen ersetzt und es werden die Nachrichtenflüsse zwischen Nutzern und konkreten Fach– bzw. Vorgangsklassen ermittelt. Bei Vorgangsklassen handelt es sich um Klassen, die speziell zur Steuerung eines Nutzungsfallsablaufs eingefügt werden. Eine Methode dieser Klasse steuert dann den Vorgangsablauf. Zu Strukturierungszwecken enthält diese Klasse eventuell noch weitere Methoden, d.h. Teile der Vorgangsmethode werden in kleinere, übersichtlichere Methoden ausgelagert.

Eine derartige interaktionsorientierte Spezifikation der Systemszenarien führt zu den Szenarien auf Nutzungsfallebene, deren Erstellung Ziel dieses Kapitels ist. Die vorausgehenden Nutzungsfälle auf Systemebene werden hierfür unter objektorientierten Gesichtspunkten nochmals genauer beschrieben, bevor das Verhalten, genauer die Interaktionen, mit dem Nutzer und zwischen den Klassen exemplarisch dargestellt wird. Sequenzdiagramme werden für diese exemplarische Darstellung verwendet und mit der Hilfe von „Highlevel Message Sequence Charts" (HMSCs) wird bereits in diesem Stadium versucht, das beispielhafte Verhalten teilweise zu vervollständigen. Hierzu erlauben die HMSCs eine Beschreibung von Alternativabläufen, von Wiederholungen (Schleifen) und parallelen Abläufen. Während z.B. bei exemplarischen Sequenzdiagrammen nur beispielhaft die Erzeugung eines Bestelleintragsobjekts gezeigt wird, kann mit den HMSCs verdeutlicht werden, daß bei größeren Bestellungen mehrere Bestellobjekte angelegt werden müssen.

Im Abschnitt 6.1 wird eine Einführung zu den Szenarien auf Nutzungsfallebene und den verwendeten Beschreibungstechniken gegeben. Der Nutzungsfall zur Bestellung wird im Abschnitt 6.2 genauer betrachtet. In den nächsten drei Abschnitten werden die Nutzungsfälle zum Buchvormerken (Abschnitt 6.3), zum Bestellen der vorgemerkten Bücher (Abschnitt 6.4) und zur Büchersuche (Abschnitt 6.5) in verschiedenen Detaillierungsgraden diskutiert. Abschnitt 6.6 liefert eine Zusammenfassung und Diskussion der Ergebnisse.

6.1 Einführung – Szenarien auf Nutzungsfallebene

Bei Szenarien auf Nutzungsfallebene wird nach [Bre98a] für einen spezifischen Nutzungsfall der abstrakte Dialog[1] des externen Akteurs mit dem System beschrieben. Darüber hinaus enthalten diese Szenarien die als Reaktionen auf die externen Nachrichten auftretenden internen Nachrichtenflüsse zwischen Objekten im System. Typischerweise bei den Szenarien involvierte Objekte sind der externe Akteur, ein dem Nutzungsfall zugeordnetes Vorgangsobjekt und Fachklassen, die vom Vorgangsobjekt angesprochen werden.

Das dynamische Verhalten zwischen den gerade beschriebenen Objekten wird exemplarisch mittels Sequenzdiagrammen dargestellt. Zur Beschreibung von Alternativen werden auf dieser Abstraktionsebene sogenannte „Highlevel Message Sequence Charts“ (HMSCs) verwendet. Diese Diagramme erlauben neben der Beschreibung von Alternativabläufen auch eine Darstellung von unendlichen Durchläufen. Im weiteren Verlauf interpretieren wir jedoch – wie bisher in Sequenzdiagrammen – alle Wiederholungen als endlich. Ein HMSC ist ein gerichteter Graph, dessen Knoten auf Sequenzdiagramme oder auf HMSCs verweisen. Der Graph beschreibt sozusagen einen „Fahrplan“, d.h. in welcher Reihenfolge die Knoten erreicht werden können.

Bei den HMSC ist noch zu erwähnen, daß die Aktivitäten nicht in den Kanten stattfinden, sondern in den Knoten, in denen sich die Sequenzdiagramme verbergen. Auch bei geschachtelten HMSCs, d.h. wenn ein Knoten auf ein weiteres HMSC verweist, gelangt man auf der untersten Ebene zu einem Sequenzdiagramm, das inhärent Aktionen beinhaltet.

Die HMSCs sind keine Beschreibungselemente der UML. Die graphische Darstellung stammt aus dem Standard MSC–96. Eine Einführung zu den HMSCs wird beispielsweise in [Krü99] beschrieben. Neben der Verwendung von HMSCs werden auch Erweiterungen in Sequenzdiagrammen angewandt, wie Schleifen oder parallele Abläufe, die eine präzisere Darstellung in Sequenzdiagrammen erlauben und die Ausdruckskraft verstärken. Diese Erweiterungen sind in [BBH+99] erläutert.

Bevor das exemplarische Verhalten in der oben beschriebenen Art und Weise ermittelt werden kann, müssen die Nutzungsfälle auf der Ebene des Systems aus Kapitel 4 genauer spezifiziert werden. In dieser objektorientierten Spezifikation findet eine Analyse des Akteurs, der Ein– und Ausgabedaten, der veränderten Fachklassen und des Verhaltens mit

[1] Hier wird von einem abstrakten Dialog gesprochen, da zu diesem Zeitpunkt noch keine Angaben über die Eingabeform (Buttons, Textfelder, ...) getroffen wurden. Auch beinhaltet eine eingezeichnete Eingabenachricht oftmals bei genauer Analyse mehrere Einzeleingaben des Nutzers.

Varianten statt. Die Beziehung zu anderen Nutzungsfällen wird in sogenannten Nutzungsfalldiagrammen (engl. „Use Case Diagrams“) dargestellt. Sie sollen vor allem das Zusammenspiel mit anderen Nutzungsfällen und inhärente Teilaufgaben des Systems graphisch repräsentieren und sind ein Teil der Unified Modeling Language.

Die in den folgenden Fällen verwendete textuelle Beschreibung zur objektorientierten Spezifikation von Nutzungsfällen orientiert sich an den die in der Habilitationsschrift von Ruth Breu [Bre98a] vorgestellten Darstellungen. Die Verwendung von Nutzungsfalldiagrammen wird in [BRJ98] [BRS97], [FS98], [Jac93] und [JBR98] erläutert.

Die Verwendung von Sequenzdiagrammen zur exemplarischen Darstellung von Interaktionen wird in [BBH+99], [BGH+97a], [Bre98a], [Bre98b], [BRJ98], [BRS97], [Bur97], [FS98], [JBR98] und [Oes97] beschrieben.

6.2 Die Bestellung

Eine Bestellung eines Buches durch einen Kunden setzt sich aus der Buchermittlung, der Selektion zum Bestellen und dem Aktivieren der Bestellung zusammen. Die Bestellung soll in diesem Abschnitt ausgehend vom Systemszenario und der Pflichtenheftfunktion genauer analysiert werden. Wie sich bei genauerer Betrachtung des Systemszenarios zur Bestellung herausstellt, verwendet es nur weitere Nutzungsfälle und enthält keine eigenen Interaktionen (vgl. Abbildung 6.1 auf der nächsten Seite). Aus diesem Grund werden in diesem Abschnitt nur HMSCs erstellt. Diese sollen die Ablauffolge der in den Knoten referenzierten Sequenzdiagramme bzw. HMSCs beschreiben.

6.2.1 Vorausgehende Spezifikation

Der Nutzungsfall zur Bestellung steuert den Ablauf des Bestellvorgangs und dient zur Ablaufsteuerung für die Produktfunktionen */F:Suche/*, */F:Blättern/*, */F:Vormerken/* und */F:Bestellung/* (siehe Seite 12). Bei der Identifikation der Nutzungsfälle zur Kundensitzung (siehe Abschnitt 5.2 auf Seite 56) wurde aus diesen Funktionsanforderungen bereits das in Abbildung 6.1 dargestellte Systemszenario zur Bestellung ermittelt.

Wie diese Abbildung zeigt, können vor einer Bestellungseinleitung mehrere Bücher gesucht und zum Kauf vorgemerkt werden. Nach der Bestellauswahl kann die definitive Bestellung eingeleitet werden, bei der persönliche und lieferspezifische Daten angegeben werden. Das Sequenzdiagramm auf Systemebene zeigt hierzu nur die notwendigen Stellvertreter und den zugehörigen Iterationsoperator.

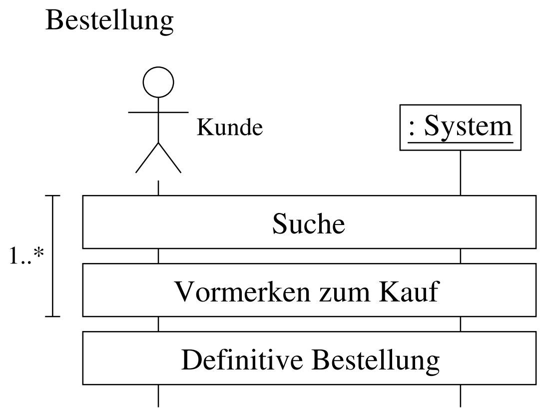


Abbildung 6.1: Systemszenario: Bestellung

6.2.2 HMSC zur Bestellung

Da sich hinter den Stellvertretern im Systemszenario zur Bestellung (siehe Abbildung 6.1) weitere Nutzungsfälle verbergen, die entweder als Sequenzdiagramme oder als Highlevel Message Sequence Charts dargestellt werden können, erscheint die Darstellung des Bestellvorgangs als HMSC sinnvoll. Abbildung 6.2 zeigt dieses Diagramm.

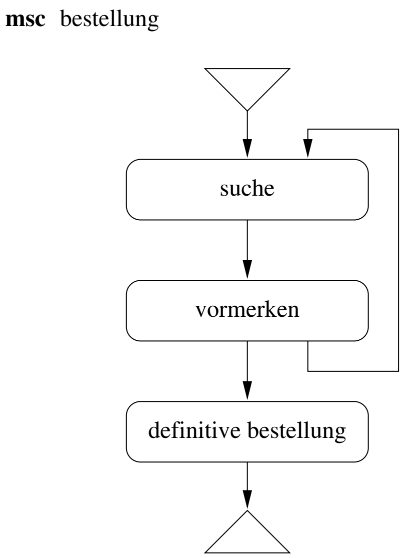


Abbildung 6.2: HMSC zur Bestellung

Die Stellvertreter wurden als Knoten übernommen und zwischen den Start– und Endknoten der lineare Ablauf eingefügt. Der Iterationsoperator wurde als gerichtete Kante zwischen den Knoten „vormerken“ und „suche“ in das HMSC aufgenommen, so daß generell beliebig viele Bücher ausgewählt und vorgemerkt werden können. Die Diagramme aus Abbildung 6.1 und 6.2 sind in diesem Fall, bis auf die Tatsache, daß aus dem Sequenzdiagramm weitere Sequenzdiagramme referenziert werden müssen, während beim HMSC sowohl HMSCs als auch Sequenzdiagramme referenziert werden können, semantisch äquivalent.

6.2.3 HMSC zur Suche

Die Suche weist ebenfalls wie die Bestellung keine eigenen Interaktionen auf, da sie nur einen Stellvertreter für die Büchersuche bzw. für das Blättern im Buchkatalog besitzt, dem der Iterationsoperator mit der Eigenschaft „mindestens einmal“ zugewiesen wurde (siehe Abbildung 5.2). Zwar handelt es sich bei dem Suchvorgang um ein eigenes Systemszenario, aber neben der Entwicklung eines HMSC ist hierbei keine weitere Spezifikation notwendig. Aus diesem Grund wird das zugehörige HMSC in diesem Abschnitt zur Spezifikation der Bestellung mit aufgenommen. Abbildung 6.3 zeigt das HMSC zum Systemszenario Suche.

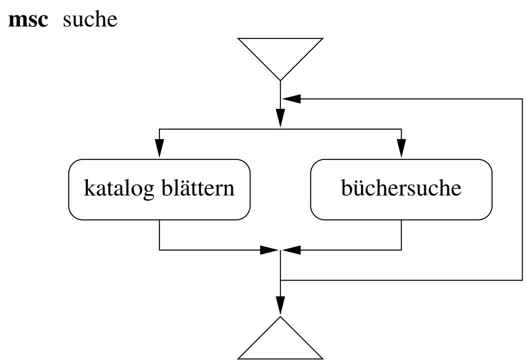


Abbildung 6.3: HMSC zur Suche

Die Alternativen „Büchersuche“ und „Blättern im Katalog“ im Stellvertreter zum Systemszenario Suche ergeben im HMSC zwei Pfade mit jeweils einem entsprechenden Knoten. Der Sequenziterator fordert wieder eine zusätzliche gerichtete Kante. Nachdem die Suche mindestens einmal als konkrete Büchersuche oder als Blättern im Katalog durchgeführt wurde, kann sie beliebig oft wiederholt werden.

6.2.4 Nachbemerkung

Aus den Systemszenarien Bestellung und Suche wurden in diesem Abschnitt zwei Highlevel Message Sequence Charts erstellt. Da sowohl die Suche als auch die Bestellung nur weitere Nutzungsfälle aufrufen, entweder sequentiell oder alternativ, eignen sich hier die HMSCs zur Darstellung. Die Knoten in den HMSCs referenzieren entweder die Sequenzdiagramme zu den entsprechenden Nutzungsfällen oder weitere HMSCs, die sich letztendlich ebenfalls auf Sequenzdiagramme abstützen.

Die Iterationsoperatoren aus den Systemszenarien bilden sich auf gerichtet Kanten in HMSCs ab. Dem Standard MSC–96 entsprechend erhalten HMSCs einen Bezeichner, der hinter dem Wortsymbol „msc“ angefügt wird.

Die Erzeugung eines exemplarischen Sequenzdiagramms ist bei der Bestellung nicht notwendig, da neben den Stellvertretern keine Interaktionen zwischen Akteur und System ersichtlich sind, so daß keine internen Nachrichtenflüsse im System bzw. zwischen Akteur und System ermittelt werden können.

Die in den Systemszenarien referenzierten Nutzungsfälle „Vormerken zum Kauf“, „Definitive Bestellung“ und „Büchersuche“, die den HMSC–Referenzen „vormerken“, „definitive bestellung“ und „büchersuche“ entsprechen, werden in den Abschnitten 6.3 bis 6.5 besprochen. Die zweite Suchmöglichkeit, das Blättern im Katalog, wird im Rahmen dieser Arbeit nicht näher betrachtet.

6.3 Vormerken zum Kauf

Nachdem ein Buch oder mehrere Bücher gefunden und ausgewählt wurden, können sie zum Kauf vorgemerkt werden. Sie werden hierzu in einen virtuellen Einkaufskorb abgelegt, der später bei der definitiven Bestellung herangezogen wird. Ausgehend von der Pflichtenheftfunktion und vom Systemszenario wird eine objektorientierte Spezifikation des Nutzungsfalls erstellt. Aus dieser Beschreibung wird ein exemplarisches Sequenzdiagramm erzeugt und in einem HMSC werden, falls vorhanden, Iterationen und Alternativen, ergänzt. Am Ende soll die Ablauffolge beim Vormerken zum Kauf festgelegt sein. Auf eine Ausarbeitung der in den HMSCs referenzierten Sequenzdiagramme wird hier im Rahmen dieser Arbeit verzichtet.

6.3.1 Vorausgehende Spezifikation

Aus der Produktfunktion */F:Vormerken/* wurde bereits bei der Identifikation der Nutzungsfälle im Kapitel 5 das Systemszenario „Vormerken zum Kauf“ erstellt. Abbildung 6.4 auf der nächsten Seite zeigt diesen Nutzungsfall auf Systemebene als Ausschnitt aus Abbildung 5.2 auf Seite 57.

Sequenzdiagramme können in ihrer Beschreibungstiefe variieren. Das Sequenzdiagramm zum Systemszenario für diesen Nutzungsfall verwendet nur eine einzige Nachricht vom Nutzer zum System, in der er den Auftrag zum Vormerken gibt. Jedoch steckt in dieser Nachricht inhärent die Eingabe der Exemplaranzahl für ein Buch oder mehrere Bücher. Die Bestätigung enthält ebenfalls eine Kontrollausgabe der Buchdaten.

Ausgehend von diesem Systemszenario erfolgt nun eine objektorientierte Spezifikation des Nutzungsfalls.

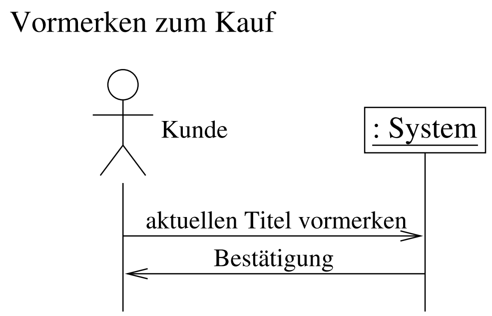


Abbildung 6.4: Systemszenario: Vormerken zum Kauf

6.3.2 Objektorientierte Spezifikation

Der Akteur, die Ein– und Ausgabedaten und die veränderten Fachklassen werden im folgenden genauer analysiert. Eine verbale Beschreibung des Nutzungsfalls mit möglichen Varianten, sowie die Erstellung eines Nutzungsfalldiagramms sollen die objektorientierte Spezifikation außerdem vervollständigen, so daß in den nächsten Teilabschnitten ein exemplarisches Sequenzdiagramm (Teilabschnitt 6.3.3) und ein Highlevel Message Sequence Chart (6.3.4) erstellt werden kann.

Akteur

Der Bookshop–Anwender ist hier der Akteur. Natürlich kann auch ein Mitglied des Bookshop–Personals in die Rolle des Akteurs als Anwender treten, und als solches z.B. Suchanfragen tätigen.

Eingabedaten

Ein einzelnes Buch bzw. mehrere auf eine Suchanfrage gefundene Bücher werden vom System beim Aufruf der Funktion Vormerken übergeben.

Weiterhin legt der Anwender für jedes vorzumerkende Buch die gewünschte Exemplaranzahl fest.

Ausgabedaten

Vor jeder Eingabe der Exemplaranzahl wird dem Nutzer der Autor, der Titel und der Preis des Buches präsentiert.

Nachdem für alle Bücher die gewünschte Anzahl festgelegt wurde, wird der ganze Einkaufskorb nochmals an den Benutzer ausgegeben. Zusätzlich wird der Gesamtpreis der im Einkaufskorb enthaltenen Elemente berechnet und ausgegeben.

Der Einkaufskorb nach dem Hinzufügen der neuen Einträge wird beim Beenden des Nutzungsfalls an das System ausgegeben.

Veränderte Fachklassen

Die Fachklasse Einkaufskorb wird um die ausgewählten Einträge mit Exemplaranzahl ergänzt.

Beschreibung

Nach dem Auswählen von Büchern durch eine Suchanfrage oder durch Blättern im Katalog können diese für eine definitive Bestellung vorgemerkt werden.

Die Ergebnisliste der Auswahl wird an die Operation zum Vormerken übergeben. Nach und nach wird jeder Listeneintrag, d.h. das Buch mit Titel, Autor und Preis, dem Kunden präsentiert und er kann die gewünschte Anzahl eingeben. Die Eingabe der Exemplaranzahl wird durch eine wiederholte Ausgabe der Buchdaten inklusive Anzahl bestätigt.

Anschließend werden die Einträge in den Einkaufskorb aufgenommen und sein Inhalt wird dem Anwender vollständig präsentiert. Der Gesamtpreis der im Einkaufskorb enthaltenen Bücher wird ermittelt und ebenfalls ausgegeben.

Variante

Wird bei der Abfrage nach der Anzahl eine Null eingegeben, so wird der entsprechende Eintrag nicht in den Einkaufskorb mit aufgenommen.

Graphische Darstellung

Abbildung 6.5 zeigt das Nutzungsfalldiagramm. Der Zusammenhang dieses Nutzungsfalls mit dem Akteur und den anderen Nutzungsfällen soll hierdurch verdeutlicht werden.

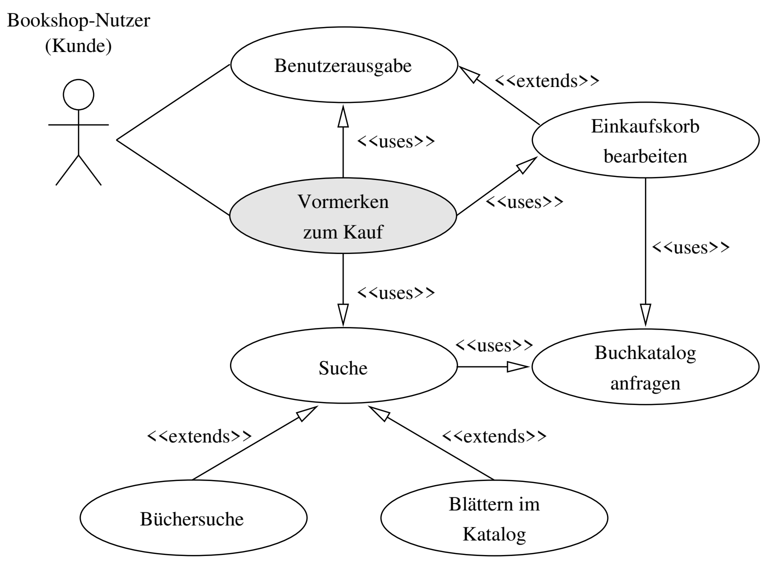


Abbildung 6.5: Nutzungsfalldiagramm: Vormerken zum Kauf

Im Mittelpunkt stehen der Akteur und der soeben beschriebene Nutzungsfall „Vormerken zum Kauf“. Das Vormerken bedient sich der vorgeschalteten Suchoperation, in dem es davon die Ausgabeliste verwendet. Bei Hinzufügen von Einträgen in den Einkaufskorb wird der Nutzungsfall „Einkaufskorb bearbeiten“ angesprochen. Die Ausgabe erfolgt bei graphischen Benutzerschnittstellen über eine gemeinsame Ausgabe, die ebenfalls beim Vormerken Verwendung findet. Der Nutzungsfall Benutzerausgabe muß hierbei den auszugebenden Text formatieren. Die Suche und die Einkaufskorbbearbeitung verwenden daneben noch den Nutzungsfall Buchkatalog anfragen. Da alle diese Nutzungsfälle in der beschriebenen Art und Weise andere verwenden, handelt es sich hierbei um sogenannte USES–Beziehungen zwischen den Nutzungsfällen.

Die Nutzungsfälle Büchkatalog anfragen und das Blättern im Bücherkatalog erweitern die Suche, und Einkaufskorb bearbeiten erweitert die Benutzerausgabe, so daß es sich hierbei um EXTENDS–Beziehungen handelt. Die USES– und EXTENDS–Beziehungen lassen sich bei der Erzeugung des Nutzungsfallsdiagramms nicht systematisch aus der verbalen Spezifikation des Nutzungsfalls ableiten. Zwar können die Ein– und Ausgabedaten sowie der Akteur in das Diagramm eingebracht werden, aber eine allgemeingültige Transformationregel konnte hier nicht festgestellt werden. Dies liegt zum Teil auch daran, daß die Interpretation der Nutzungsfalldiagramme nicht eindeutig ist und daß Nutzungsfalldiagramme nie vollständig dargestellt werden können. In diesem Diagramm stehen die EXTENDS–Beziehungen für Verhaltensverfeinerungen und die USES–Beziehungen für das Anwenden eines Teilverhaltens eines anderen Nutzungsfalls.

6.3.3 Exemplarische Verhaltensspezifikation

Aus der objektorientierten Spezifikation und dem Systemszenario für das Buchvormerken wird in diesem Teilabschnitt ein exemplarisches Sequenzdiagramm erzeugt. Die Klasse System aus dem Systemszenario wird in die Fachklassen und in ein Vorgangsobjekt aufgebrochen, so daß die systeminternen Nachrichtenflüsse beschrieben werden können. Die externen Nachrichten werden detaillierter dargestellt als auf Systemebene. Abbildung 6.6 auf der nächsten Seite zeigt das Sequenzdiagramm für dieses Nutzungsfall.

Das Vorgangsobjekt VgVormerken wurde neu hinzugefügt und steuert den Ablauf. Zu den Markierungen sind noch folgende Anmerkungen zu ergänzen:

Punkt 1: Das Sitzungsobjekt, welches den Ablauf der Kundensitzung steuert, erzeugt das Vorgangsobjekt und übergibt als Parameter das Ausgabeterminal und die Liste mit den ausgewählten Büchern.

Punkt 2: Die Buchdaten werden angezeigt. Die gewünschte Anzahl wird vom Kunden eingetragen.

Als Alternative darf ein Buch mit der Exemplaranzahl kleiner Eins nicht in den Einkaufskorb aufgenommen werden.

Im Beispiel enthält die Liste nur ein einziges Element. Alternativ könnten hier auch mehrere Elemente übergeben werden.

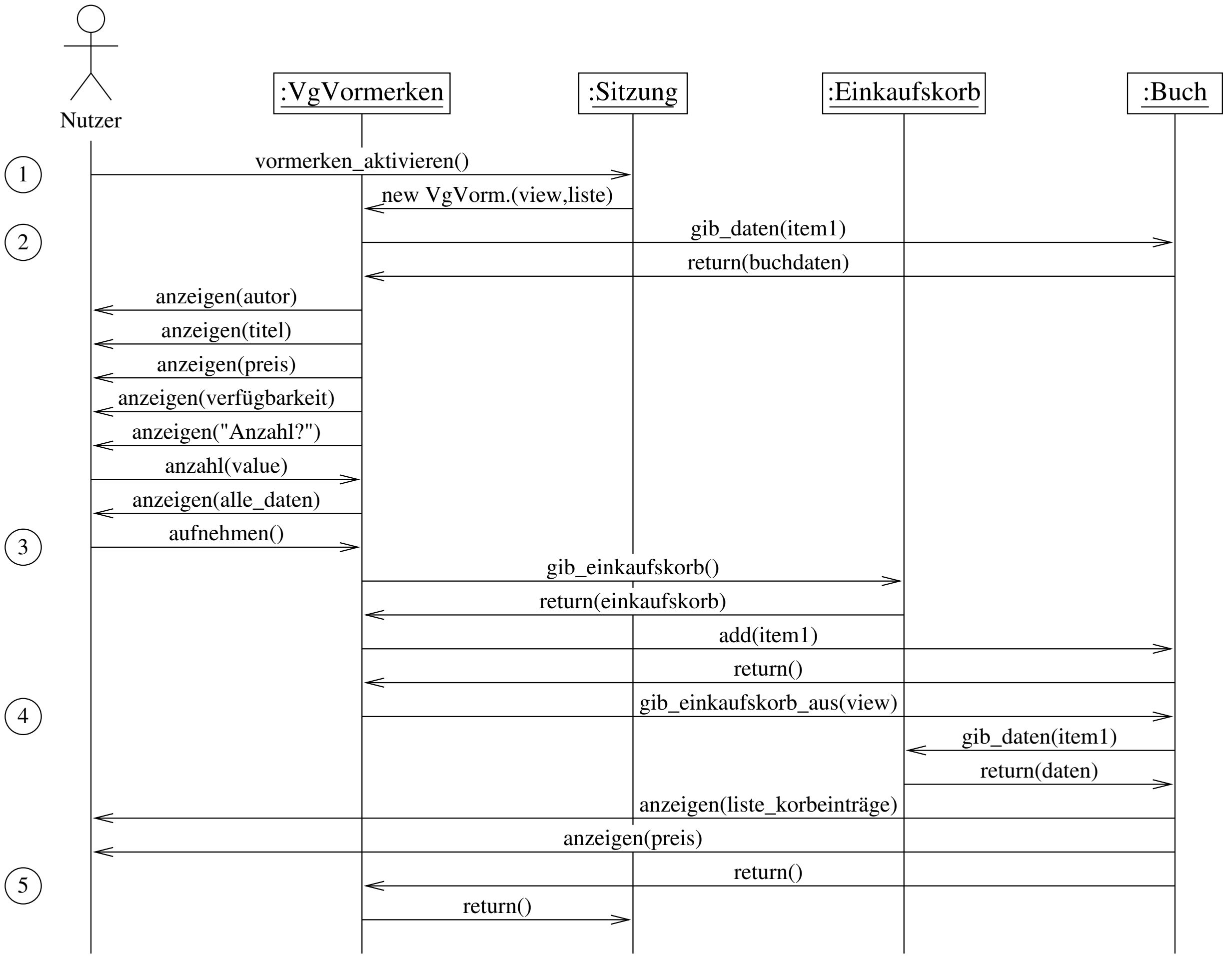


Abbildung 6.6: Sequenzdiagramm zum Nutzungsfall: Vormerken zum Kauf

Punkt 3: Nach der Aufnahmebestätigung wird der Einkaufskorb vom Vorgangsobjekt ermittelt und das selektierte Buch hinzugefügt. Alternativ muß bei mehreren Listeneinträgen jeder Eintrag in den Einkaufskorb hinzugefügt werden.

Punkt 4: Der Inhalt des Einkaufskorbs mit dem Gesamtpreis wird angezeigt, hier exemplarisch nur für ein Buch. Bei mehreren Büchern müßten die Buchdaten für jedes Buch einzeln angefordert werden.

Punkt 5: Die Kontrolle wird wieder an das Sitzungsobjekt zurückgegeben.

Die hier aufgezählten Varianten werden im nächsten Teilabschnitt, bei der Ermittlung des HMSC, zur Erzeugung von Abzweigungen bzw. Schleifen eingesetzt.

6.3.4 Highlevel MSC

Das eben entwickelte exemplarische Sequenzdiagramm wird in kleine Einheiten von unabhängigen Interaktionen aufgesplittet. Das HMSC in Abbildung 6.7 zeigt die Knoten, die dann auf die Diagrammeinheiten verweisen. Zwischen dem Start– und Endknoten wurde wieder der sequentielle Ablauf eingetragen.

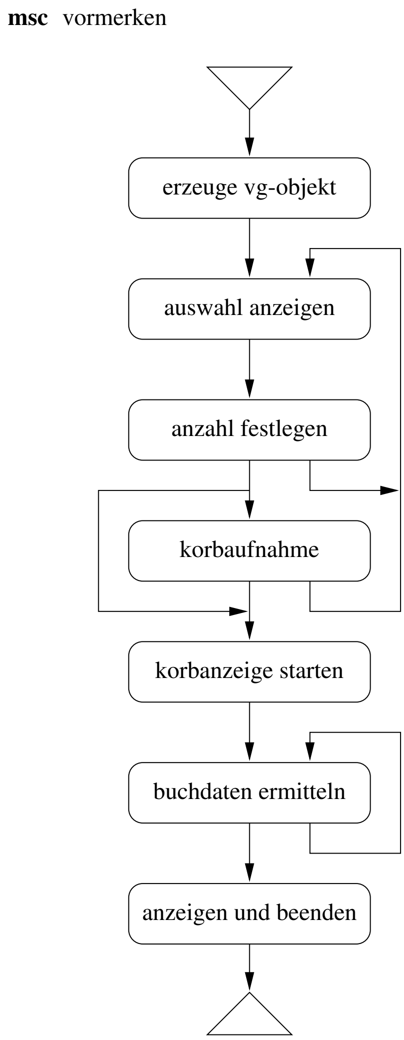


Abbildung 6.7: HMSC zum Vormerken von Büchern im Einkaufskorb

Die Alternativen in der Anmerkungsbeschreibung zum exemplarischen Sequenzdiagramm bringen noch folgende Ergänzungen in das HMSC zum Buchvormerken ein:

- Falls bei der Suche mehrere Bücher zur Einkaufskorbablage ermittelt wurden, müssen die Diagrammeinheiten zur „auswahl anzeigen“, „anzahl festlegen“ und „korbaufnahme“ mehrfach durchlaufen werden. Es wird hierzu eine gerichtete Kante von „korbaufnahme“ zu „auswahl anzeigen“ eingeführt.

- Wird die Korbanzahl kleiner Eins gewählt, so findet die Korbaufnahme nicht statt. Beim letzten Eintrag wird dann mit der Korbanzeige fortgesetzt, ansonsten mit dem Anzeigen des nächsten Buches. Die zwei gerichteten Kanten „anzahl festlegen“ – „korbanzeige starten“ und „anzahl festlegen“ – „auswahl anzeigen“ werden im HMSC ergänzt.

- Die Buchdaten müssen für jedes Buch ermittelt werden. Eine Schleife wird beim „buchdaten ermitteln“ hierzu benötigt.

Die Sequenzdiagramme, die von den Knoten im HMSC referenziert werden, sind in diesem Abschnitt nicht explizit dargestellt. Das exemplarische Sequenzdiagramm aus Abbildung 6.6 auf Seite 86 muß hierzu nur an den geeigneten Stellen unterteilt werden. Eine solche Aufteilung wird im Abschnitt 6.4 an einem Beispiel gezeigt.

6.3.5 Nachbemerkung

Ausgehend von der Pflichtenheftfunktion und vom Systemszenario wurden die objektorientierte Spezifikation, ein exemplarisches Sequenzdiagramm und ein HMSC erstellt.

Die objektorientierte Spezifikation in ihrer verbalen Art und Weise dient einer genaueren Beschreibung des Nutzungsfalls. Die veränderten Fachklassen aus dem Anwendungskern (siehe Kapitel 4) wurden neben den Aus– und Eingabedaten, dem Akteur, einer Beschreibung und einer weiteren Alternativbeschreibung ermittelt. Das Nutzungsfalldiagramm zeigt die Verbindung und Erweiterung zu anderen Nutzungsfällen. Nutzungsfalldiagramme sind in gewisser Weise unvollständig, denn es könnten auch noch weitere Nutzungsfälle aus dem System referenziert werden. Sie wurden hier aber zum Ermitteln der „näheren Umgebung“ des zu beschreibenden Nutzungsfalls verwendet und der gewählte Detaillierungsgrad reicht hierzu aus.

Aus der objektorientierten Spezifikation und der zusätzlichen Miteinbeziehung der vorausgehenden Spezifikation wurde das exemplarische Sequenzdiagramm erzeugt. Es zeigt einen beispielhaften Ablauf, bei dem keine Schwierigkeiten auftreten. Solche Abläufe werden auch als Sunny–Day–Szenarien bezeichnet.

Die möglichen Abläufe eines Rainy–Days wurden im HMSC ergänzt. Diese Diagrammart erlaubt Alternativabläufe und Wiederholungen, die hiermit sehr intuitiv dargestellt werden können. Es werden hierzu zum normalen Ablauf Verzweigungen und gerichtete Kanten eingefügt. Die Alternativabläufe wurden aus der objektorientierten Spezifikation zuerst in die verbale Beschreibung zum Sequenzdiagramm aufgenommen und von dort in das HMSC transformiert.

Eine mögliche Erweiterung zu den HMSCs wäre die Anbringung von Bedingungen an die Kanten. So könnten Alternativabläufe aus dem HMSC ohne Betrachtung der weiteren geschachtelten HMSCs oder der referenzierten Sequenzdiagramme abgelesen werden. Zähler könnten die Durchlaufanzahl einer Schleife begrenzen.

6.4 Bestellung der im Einkaufskorb liegenden Bücher

Nachdem im letzten Abschnitt die Bücher in einem virtuellen Einkaufskorb abgelegt wurden, kann jetzt zur virtuellen Kasse gegangen werden. Beim Internet–Bookshop bedeutet dies die definitive Aufgabe der Bestellung. Aus dem Systemszenario und aus der Pflichtenheftbeschreibung soll zuerst ein exemplarischer Ablauf ermittelt werden. Weiterhin soll dieser Ablauf mit Alternativen und Sequenzen angereichert werden, so daß letztendlich der Nutzungsfall mit seinem Umfeld im nötigen Detaillierungsgrad beschrieben ist.

Die genaue Beschreibung führt wieder zu einer objektorientierten Spezifikation inklusive Nutzungsfalldiagramm (Teilabschnitt 6.4.2). Der exemplarische Ablauf wird in einem Sequenzdiagramm dargestellt und erläutert (Teilabschnitt 6.4.3). Hierbei treten Ergänzungen am Fachklassendiagramm auf, wie im Teilabschnitt 6.4.4 beschrieben wird. An dem Sequenzdiagramm werden im Teilabschnitt 6.4.5 noch einige Verfeinerungen des Ablaufs vorgenommen. Teilabschnitt 6.4.6 befaßt sich mit HMSCs und der zugehörigen Aufteilung von Sequenzdiagrammen. Eine mögliche Parallelisierung hat Teilabschnitt 6.4.7 zum Thema.

6.4.1 Vorausgehende Spezifikation

Der Nutzungsfall zum Bestellen der im Einkaufskorb liegenden Bücher leitet sich aus der Pflichtenheftfunktion */F:Bestellung/* auf Seite 13 ab. Hierzu wurde bereits im Abschnitt 5.2 auf Seite 56 das folgende Sequenzdiagramm (Abbildung 6.8) ermittelt, welches dieses Szenario auf Systemebene beschreibt.

Hinter den Nachrichtenflüssen zwischen Nutzer und System verbergen sich auch hier weitere Einzelnachrichten, die im Diagramm zum Systemszenario noch nicht erfaßt wurden. Die Nutzer–Interaktionen mit dem System werden neben den systeminternen Aktionen bei der Erstellung des Sequenzdiagramms im Teilabschnitt 6.4.3 genauer analysiert.

6.4.2 Objektorientierte Spezifikation

Der noch ungenau spezifizierte Nutzungsfall wird im folgenden auf verbaler Ebene unter objektorientierten Gesichtspunkten eingehend spezifiziert. Der Zusammenhang mit der Struktur des Anwendungskerns aus Kapitel 4 wird bei der Beschreibung der veränderten Fachklassen hergestellt, während die Kapitel 3 und 5 bereits bei der vorausgehenden Spezifikation im Teilabschnitt 6.4.1 verwendet wurden.

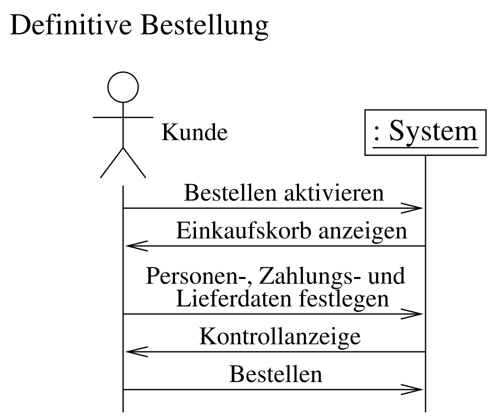


Abbildung 6.8: Systemszenario: Definitive Bestellung der im Einkaufskorb liegenden Bücher

Akteur

Die Rolle des Akteurs fällt hier auf den Bookshop–Benutzer, der als Käufer einerseits Daten eingibt und andererseits bereits im System vorhandene kontrolliert bzw. modifiziert.

Eingabedaten

Die Personendaten, Angaben zur Zahlungsweise und zur Lieferung sowie die Liste der im Einkaufskorb liegenden Bücher nach einer eventuellen Veränderung dienen als Eingabedaten.

Name und Anschrift, Land, Telefonnummer für eventuelle Rückfragen und die Email–Adresse müssen als Personendaten angegeben werden. Optional kann eine separate Rechnungsanschrift vermerkt werden.

Bei der Zahlungsweise kann der Anwender zwischen Kreditkartenbezahlung, Scheckbezahlung, Bankeinzug oder einer E–Commerce–Bezahlung auswählen.

Aus einer Liste von Paketdiensten inklusive der Post kann der Benutzer einen Lieferservice bestimmen. Außerdem besteht noch die Möglichkeit, eine Buchsendung an eine spezielle Lieferadresse zu senden.

Die systeminterne Einkaufskorbliste wird an den Nutzungsfall zum definitiven Bestellen weitergegeben.

Ausgabedaten

Während der Bearbeitung werden der Einkaufskorb inklusive Gesamtpreis, die eingegebenen Kunden–, Bezahlungs– und Lieferdaten zu Kontrollzwecken und Bestätigungen vom System ausgegeben. Die neue Bestellung mit ihren Bestelleinträgen wird systemintern nach der Bearbeitung weitergereicht.

Veränderte Fachklassen

Folgende Fachklassen werden in diesem Nutzungsfall verändert: Einkaufskorb, Bankverbindung, Kunde, Bestellung, Bestelleintrag.

Beschreibung

Nachdem der Käufer seine Bücher ausgewählt hat, wechselt er zur Bestellung. Hier wird die Bücherliste des Einkaufskorbs mit den gewünschten Exemplaranzahlen und dem Gesamtpreis noch einmal ausgegeben. Die Exemplaranzahlen können noch einmal verändert werden. Eine Reduzierung der Anzahl zu Null entspricht dem Löschen eines Buches aus dem Einkaufskorb. Dadurch bekommt der Benutzer in diesem Stadium nochmal die Möglichkeit, bestimmte Bücher aus der Liste zu entfernen.

Im folgenden gibt er die für die Bestellung notwendigen Daten ein. Die oben beschriebenen Personendaten, den Lieferservice und die benötigten Daten für die jeweilige Zahlungsweise müssen vom Käufer in einer Bildschirmmaske ausgefüllt werden. Bei einer Kreditkartenbezahlung wird die Kreditkartennummer und das Gültigkeitsdatum eingetragen, bei einem Bankeinzug Bankleitzahl und Kontonummer.

Wurden alle Eingaben erfolgreich eingegeben, werden dem Benutzer die Daten nochmals zur Kontrolle präsentiert und mit der anschließenden Bestätigung erfolgt die Übertragung der Einkaufskorbliste in die Bestelliste.

Eine besondere Betrachtungsweise muß hier den Zahlungsdaten zukommen. Da Internet–Daten entsprechend einer Postkarte gelesen werden können, müssen z.B. die Kreditkartennummer in Verbindung mit Name und Gültigkeitsdatum verschlüsselt übertragen werden, um dem Datenschutz gerecht zu werden. Vergleiche auch hierzu die Produktleistung */L:Sichere Speicherung/* auf Seite 16.

Variante 1

Die Bestellung wird nicht von einer Einzelperson, sondern von einer Firma vorgenommen. In diesem Fall müssen Firmendaten und die Daten einer Kontaktperson gespeichert werden.

Variante 2

Der Kunde wünscht eine andere Lieferadresse als seine persönliche Adresse. Dies kann z.B. bei Geschenksendungen notwendig sein. Die Eingabe einer solchen Lieferadresse soll ebenfalls ermöglicht werden.

Variante 3

Bei der Bestellung handelt es sich bereits um eine wiederholte Bestellung. Falls die Angaben zur Zahlungsweise und/oder die Personendaten gleich geblieben sind, müssen diese nicht erneut eingegeben und übertragen werden. Aus diesem Grund erhält ein Kunde nach der erstmaligen Angabe oder nach der Änderung seiner Daten die Möglichkeit einer Paßworteingabe. Zusammen mit der Email–Adresse, die als Benutzerkennwort dient, und diesem Paßwort kann sich der Käufer in Zukunft authentifizieren. Bei der Übertragung des Paßworts muß ebenfalls aus Sicherheitsgründen eine Verschlüsselung stattfinden.

Graphische Darstellung

In Abbildung 6.9 ist das zugehörige Nutzungsfalldiagramm dargestellt. Hierbei soll wieder der Zusammenhang zu anderen Nutzungsfällen und zum Akteur verdeutlicht werden.

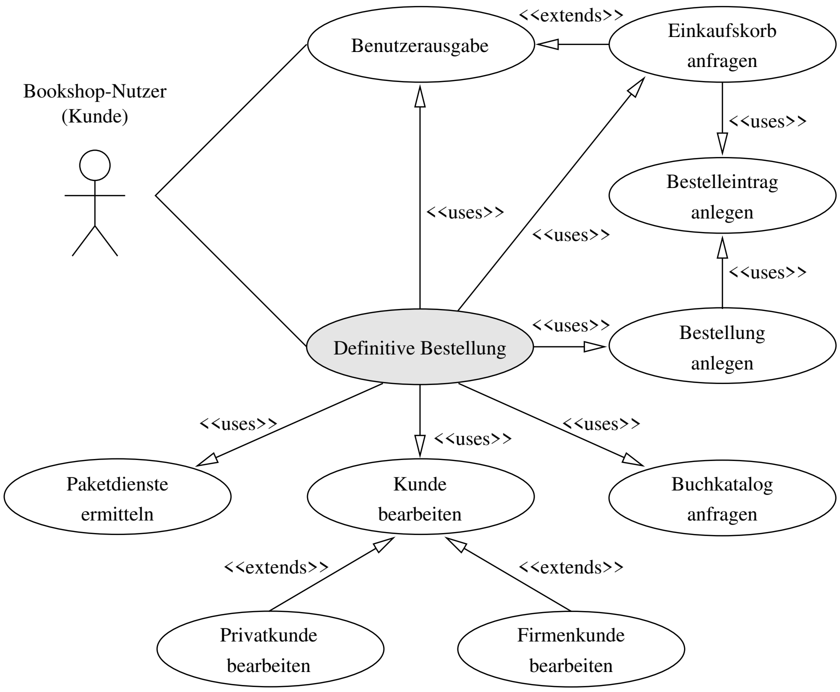


Abbildung 6.9: Nutzungsfalldiagramm: Bestellung der im Einkaufskorb liegenden Bücher

Der Anwender als Akteur empfängt Aktionen über die Benutzerausgabe und gibt sie über den aktuellen Nutzungsfall an das System.

Als weitere Nutzungsfälle verwendet der Nutzungsfall zur definitiven Bestellung den zum „Kunden bearbeiten“, zum „Einkaufskorb anfragen“, zum „Bestellung anlegen“, zum „Paketdienste ermitteln“, und den zum „Buchkatalog anfragen“. „Bestellung anlegen“ und „Einkaufskorb abfragen“ verwenden wiederum „Bestelleintrag anlegen“. Hierbei handelt es sich um USES–Beziehungen.

„Einkaufskorb anfragen“ erweitert die Benutzerausgabe und „Privatkunden bearbeiten“ bzw. “Firmenkunden bearbeiten“ ergänzen „Kunden bearbeiten“ mit der sogenannten EXTENDS–Beziehung.

Alternativ könnte der Nutzungsfall zur definitiven Bestellung auch die Benutzerausgabe erweitern, anstatt sie zu verwenden. Dies würde zu einer Änderung der USES–Beziehung in eine EXTENDS–Beziehung führen. Auf die Anwendung der USES– und EXTENDS–Beziehungen wird auf die Beschreibung zum Nutzungsfalldiagramm beim Buchvormerken auf Seite 84 verwiesen.

6.4.3 Exemplarische Verhaltensspezifikation

Die Sequenzdiagramme in den Abbildungen 6.10 und 6.11 zeigen ein exemplarisches Szenario für den Nutzungsfall „Bestellung der im Einkaufskorb liegenden Bücher". Es beschreibt den Fall, in dem ein Privatkunde die zuvor angewählten und im Einkaufskorb abgelegten Bücher nun definitiv bestellt. Außerdem handelt es sich bei dem Käufer um einen Neukunden, so daß seine persönlichen Daten im System noch nicht bekannt sind. Er bevorzugt eine Kreditkartenbezahlung und übergibt eine spezielle Lieferadresse, d.h. die Ware wird zu einem von der Wohnadresse abweichenden Ort gesendet.

Die Fachklassen leiten sich aus der Struktur des Anwendungskerns ab, wie bereits in der objektorientierten Spezifikation erläutert wurde. Die neue Vorgangsklasse VgDefBestellung steuert den gesamten Ablauf, indem sie die Fach– und Verwalterklassen anspricht. Zu dem exemplarischen Ablauf im Sequenzdiagramm sind noch folgende Anmerkungen hinzuzufügen:

Punkt 1: Den bisherigen Ablauf verwaltet das Sitzungsobjekt. Vom Nutzer, der in diesem Fall dem Bookshop–Kunden entspricht, kommt nun die externe Nachricht, die definitive Bestellung zu aktivieren.

Das Sitzungsobjekt erzeugt ein neues Vorgangsobjekt VgDefBestellung. Als Parameter wird das Ausgabeterminal zum Anzeigen der Benutzerinformation mitgegeben.

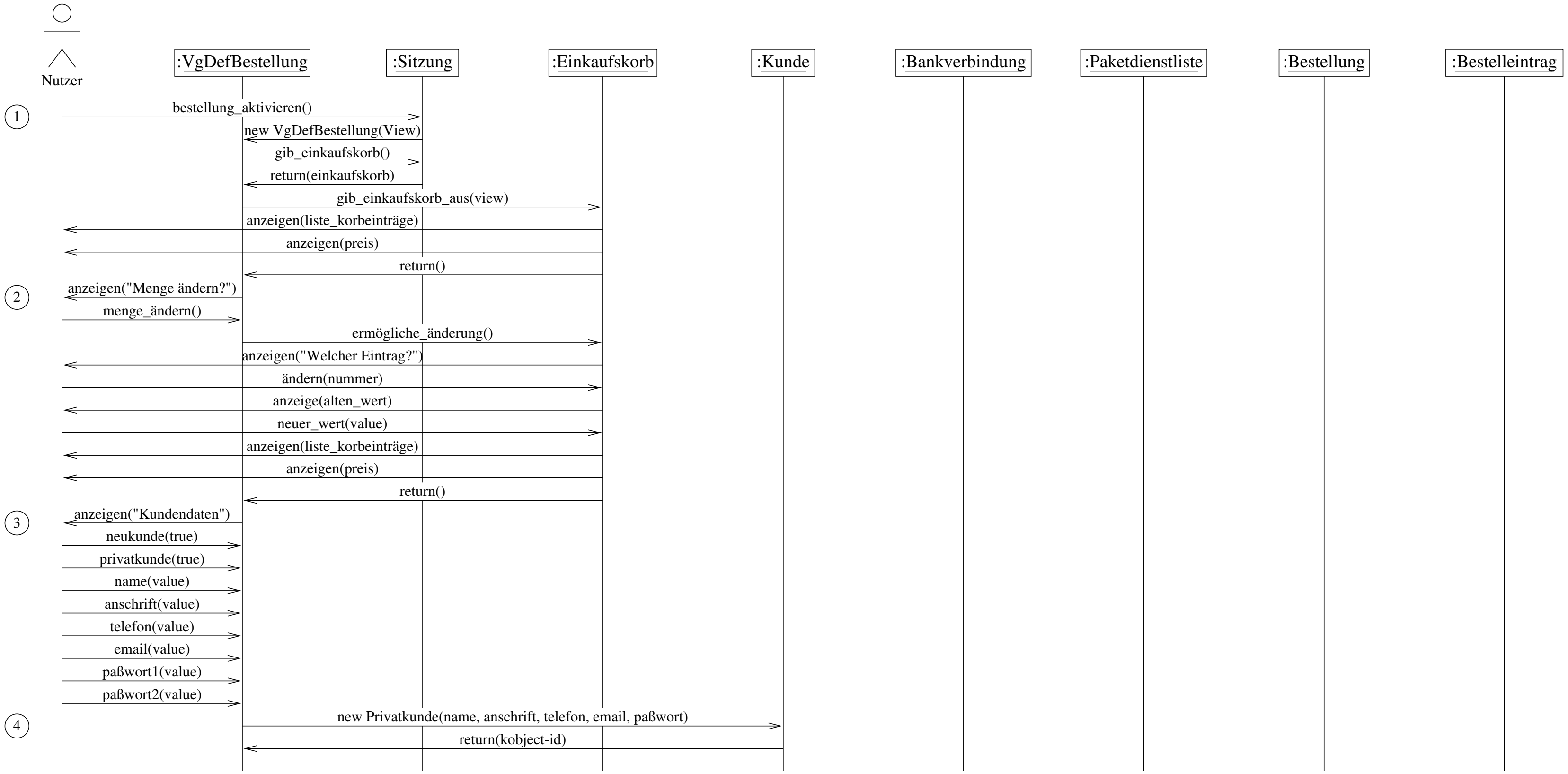
Das Vorgangsobjekt fordert von dem Sitzungsobjekt eine Referenz des Einkaufskorbobjekts an, um mit diesem in Kontakt treten zu können. Als Alternativlösung hätte auch bereits bei der Vorgangsobjekt–Erzeugung eine Parameterübergabe des Einkaufskorbs stattfinden können.

Der Einkaufskorb wird beauftragt, seine Liste an die Terminal–Ausgabe zu senden. Der Einkaufspreis soll ermittelt und ebenfalls dem Kunden präsentiert werden. Diese Interaktion könnte noch verfeinert werden, da die Buchdaten der einzelnen Buchobjekte angefordert werden müssen.

Punkt 2: Der Kunde erhält die Möglichkeit, von einzelnen Bestellartikeln die Anzahl zu ändern. So kann er z.B. die Anzahl der Exemplare eines Buches ändern oder aber auch ein Buch dadurch aus der Liste entfernen, daß er die gewünschte Anzahl auf Null setzt.

Realisiert wird eine Änderung dadurch, daß der Benutzer den zu ändernden Eintrag auswählt und die Anzahl modifiziert. Die Liste und der Gesamtpreis werden darauf wieder vollständig neu angezeigt.

Im exemplarischen Sequenzdiagramm ist nur das Ändern eines Eintrags beschrieben. Der Änderungsvorgang kann jedoch beliebig oft wiederholt oder ganz weggelassen werden.



Fortsetzung auf der nächsten Seite!

Abbildung 6.10: Sequenzdiagramm zum Nutzungsfall: Bestellung der im Einkaufskorb liegenden Bücher (Teil 1)

Abbildung 6.11: Sequenzdiagramm zum Nutzungsfall: Bestellung der im Einkaufskorb liegenden Bücher (Teil 2)

Punkt 3: Das Vorgangsobjekt fordert den Kunden auf, seine kundenspezifischen Daten anzugeben. Bei dem Käufer handelt es sich um einen Neukunden, so daß alle Daten ermittelt werden müssen. Name, Anschrift, Telefonnummer, Email–Adresse und Paßwort mit nochmaliger Bestätigung werden übergeben.

Das Sequenzdiagramm beschreibt nur den Fall des Neukunden. Bei einem Folgeeinkauf sind nur die Email–Adresse und das Paßwort anzugeben, so daß das Kundenobjekt ermittelt werden kann.

Auch die Angabe eines Firmenkunden wird nicht beschrieben. Hier sind zuerst die firmenspezifischen Daten anzugeben, gefolgt von den Daten der Kontaktperson.

Punkt 4: Das Vorgangsobjekt legt mit den gerade ermittelten Kundendaten ein neues Privatkundenobjekt an.

Da die Klasse Privatkunde von der Klasse Kunde abgeleitet wurde, kann nun ein Privatkundenobjekt als Kundenobjekt instantiiert werden.

Das neue Kundenobjekt wird aber noch nicht endgültig in die Kundenliste aufgenommen, da die Daten zuvor noch vom Händlerpersonal kontrolliert werden müssen. Beim Erzeugen des Kundenobjekts muß deshalb ein Attribut z.B. auf den Status „noch nicht übernommen“ gesetzt werden (vgl. Abbildung 5.5 auf Seite 61, Teilbild g: Kundenverwaltung als Funktion der Händler–Sitzung). Da ein solches Attribut in der Klasse Kunde noch nicht existiert, muß dieses hinzugefügt werden; die Ergänzung wird im Abschnitt 6.4.4 erläutert.

Alternativ zum Privatkunden könnte auch ein Firmenkundenobjekt angelegt werden. Hierzu wären jedoch im Punkt 3 teilweise andere Kundendaten nötig. Weiterhin wäre die Erstellung eines Kontaktpersonenobjekts notwendig.

Punkt 5: Die Zahlungsweise wird ermittelt, wobei sich der Kunde in diesem Beispiel für eine Bezahlung mit Kreditkarte entscheidet.

Als Alternativen zur Kreditkartenbezahlung wären noch der Bankeinzug, die Scheckbezahlung und eine E–Commerce–Abwicklung zu nennen.

Punkt 6: Das Vorgangsobjekt erzeugt nun ein neues Kreditkartenobjekt als Bankverbindungsobjekt.

Punkt 7: Die verfügbaren Paketdienste werden ermittelt. Über eine globale Referenz auf die Paketdienstliste fordert das Vorgangsobjekt die zur Auswahl stehenden Paketdienste an. Das Vorgangsobjekt präsentiert dem Nutzer die Liste, so daß dieser daraus einen auswählen kann.

Punkt 8: Nach einer abweichenden Lieferanschrift wird ermittelt. Da der Kunde die Ware an einen speziellen Ort geliefert bekommen will, gibt er erstens an, daß er eine abweichende Lieferanschrift verwenden will, und zweitens übermittelt er diese Anschrift an das Vorgangsobjekt.

Punkt 9: Der Käufer entscheidet sich nun endgültig zur Bestellungsausführung. Als Alternative wäre hier ein Bestellabbruch oder ein Ändern der bereits eingegebenen Daten zu nennen.

Ein neues Bestellungsobjekt wird instantiiert und mit der Kundenobjekt–Referenz, dem aktuellen Datum, der Lieferadresse, dem Lieferzuschlag und dem Paketdienst belegt.

Punkt 10: Das Einkaufskorbobjekt wird durch das Vorgangsobjekt beauftragt, seine Bestelldaten an das eben erzeugte Objekt Bestellung zu übermitteln.

Das Einkaufskorbobjekt erzeugt hierzu neue Bestelleintragsobjekte mit den notwendigen Daten. Diese sorgen dafür, daß sie in der zugehörigen Bestellung eingetragen werden. Die Kontrolle wird anschließend an das Vorgangsobjekt zurückgegeben.

Im Sequenzdiagramm ist nur exemplarisch der Eintrag eines Buches dargestellt. Das Einkaufskorbobjekt erzeugt jedoch soviele neue Bestelleintragsobjekte wie Elemente in der Einkaufskorbliste enthalten sind.

Punkt 11: Das Vorgangsobjekt fordert das neu erstellte Objekt Bestellung auf, das Gesamtgewicht und den Gesamtpreis aus den eingetragenen Bestelleinträgen zu ermitteln.

Die Invarianten aus Kapitel 4.2 auf Seite 34, die für die Klasse Bestellung eingeführt wurden, müssen hierbei eingehalten werden.

Der Einkaufskorb, aus dem die Daten in die Bestellung übernommen wurden, muß nun geleert werden.

Punkt 12: Der Käufer wird über eine erfolgreiche Ausführung der Bestellungsübernahme informiert.

Möglichkeiten, daß die Bestellungsübernahme nicht korrekt ausgeführt werden konnte, wurden im Sequenzdiagramm nicht beschrieben.

Nach Beendigung seiner Aufgaben gibt das Vorgangsobjekt die Kontrolle an das Sitzungsobjekt zurück.

6.4.4 Ergänzung im Fachklassendiagramm

Wie sich bei der Bearbeitung des Sequenzdiagrammes im letzten Teilabschnitt herausstellte, muß der Kundenklasse ein Attribut hinzugefügt werden, das Auskunft über den Kundenstatus gibt. Ein neuer Kunde muß zuerst vom Händlerpersonal geprüft werden, bevor er endgültig in die Kundendatenbank aufgenommen wird. Sollte ein Kunde versehentlich seine Anschrift falsch schreiben, so wird nach einer Überprüfung durch das Personal dieser Fehler entdeckt und eine mögliche Doppel–Speicherung vermieden. Abbildung 6.12 auf der nächsten Seite zeigt das erweiterte Fachklassendiagramm.

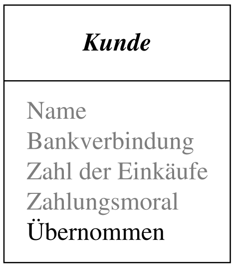


Abbildung 6.12: Ergänzung der Fachklasse Kunde

Dem neuen Attribut „Übernommen“ wird der Typ „Boolean“ zugewiesen, so daß mittels den Werten „true“ und „false“ die endgültige Aufnahme des Kunden in die Kundendatenbank überprüft werden kann.

6.4.5 Verfeinerungen des Ablaufs

In diesem Teilabschnitt wird erstens eine Darstellungsalternative zu dem im Teilabschnitt 6.4.3 erstellten exemplarischen Sequenzdiagramm gezeigt und zweitens eine Verfeinerung an dem Sequenzdiagramm durchgeführt.

- Objekte, die nicht bereits zum Startzeitpunkt des Sequenzdiagramms existieren, werden erst bei ihrer Erzeugung in das Diagramm aufgenommen. Die Lebenslinie beginnt also erst nach der Erzeugung. Bei dieser Darstellungsänderung kann das Schlüsselwort „new“ entfallen, das sonst bei der Objekterzeugung immer explizit angegeben wurde. Hierbei handelt es sich um die oben genannte Darstellungsänderung.
- In ein Sequenzdiagramm wird eine Schleife mit aufgenommen. Die auch vom SysLab–Team vorgeschlagene Darstellung bietet eine Alternative zur Darstellung von Schleifen in HMSCs. Verfeinerungstechniken zu Sequenzdiagrammen sind in [BBH+99] beschrieben.

Das Sequenzdiagramm zur definitiven Bestellung wird zur besseren Übersicht in drei Teildiagramme aufgeteilt. In jedes Teildiagramm werden nur noch die agierenden Fachklassen neben dem Nutzer und dem Vorgangsobjekt aufgenommen, so daß das Diagramm nicht mit überflüssigem Material überschwemmt wird. Die Objekterzeugungen werden in den Abbildungen 6.13 bis 6.15 gemäß der obigen Beschreibung dargestellt.

Im ersten Teil–Sequenzdiagramm in Abbildung 6.13 auf der nächsten Seite wurden die Fachklassenobjekte Bankverbindung, Paketdienste, Bestellung und Bestelleintrag mit ihren Lebenslinien entfernt, da mit ihnen in diesem Teildiagramm keine Interaktionen stattfinden. Hierbei wird davon ausgegangen, daß mit nicht eingezeichneten Objekten im Nutzungsfallabschnitt, der im Teil–Sequenzdiagramm dargestellt ist, keine Nachrichtenflüsse

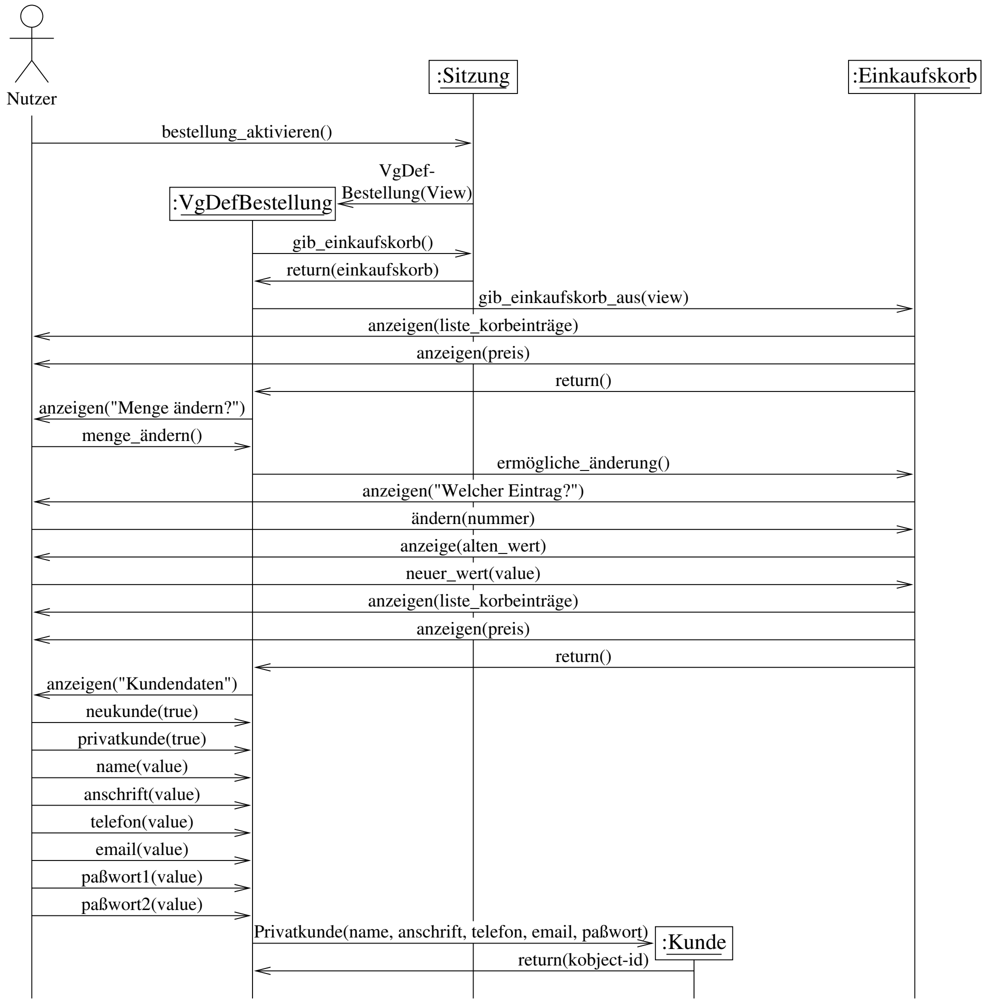


Abbildung 6.13: Sequenzdiagramm mit expliziten Objekterzeugungs–Zeitpunkten (Teil 1)

stattfinden. Eine Alternative hierzu wäre, daß beim Fehlen eines Objekts beliebige Interaktionen mit diesem zugelassen würden. Im Rahmen dieser Arbeit wird jedoch die erste Variante verwendet, d.h. nur die wirklich im Nutzungsfall eingezeichneten Nachrichtenflüsse finden statt.

Das Kundenobjekt mit seiner Lebenslinie wird erst bei seiner Erzeugung in die Abbildung aufgenommen. Das Schlüsselwort „new" bei der Erzeugungsnachricht kann in dieser Diagrammdarstellung weggelassen werden, da hier die Objekterzeugung durch das Auftreten des Objekts ersichtlich wird.

Abbildung 6.14 auf der nächsten Seite zeigt den zweiten Teil. Hier sind die Bestellungs– und die Bestelleintragsobjekte nicht eingezeichnet. Auch im Kundenobjekt, das im ersten Teil erzeugt wurde, gehen hier keine Nachrichten mehr ein, so daß dieses ebenfalls weggelassen werden kann. Das Bankverbindungsobjekt wird erst wieder zu einem späteren

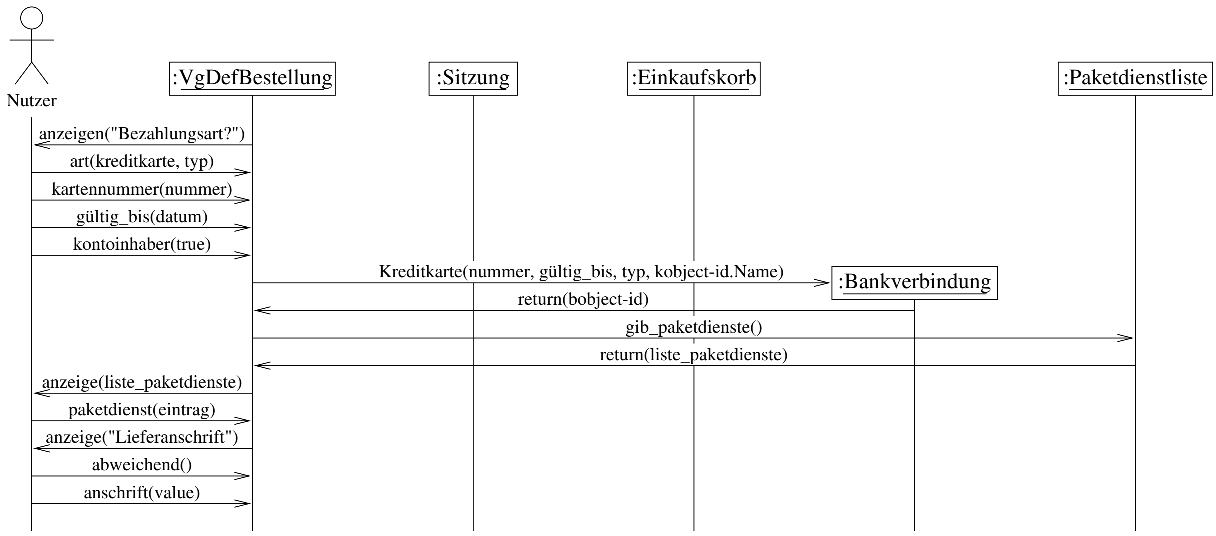


Abbildung 6.14: Sequenzdiagramm mit expliziten Objekterzeugungs–Zeitpunkten (Teil 2)

Zeitpunkt erzeugt, so daß sich die Lebenslinie für dieses Objekt nicht über das ganze Teildiagramm erstreckt und das Objektsymbol, ein Rechteck mit dem unterstrichenen Namen und vorgestelltem Doppelpunkt, wird erst beim Erzeugen des Objekts gezeichnet.

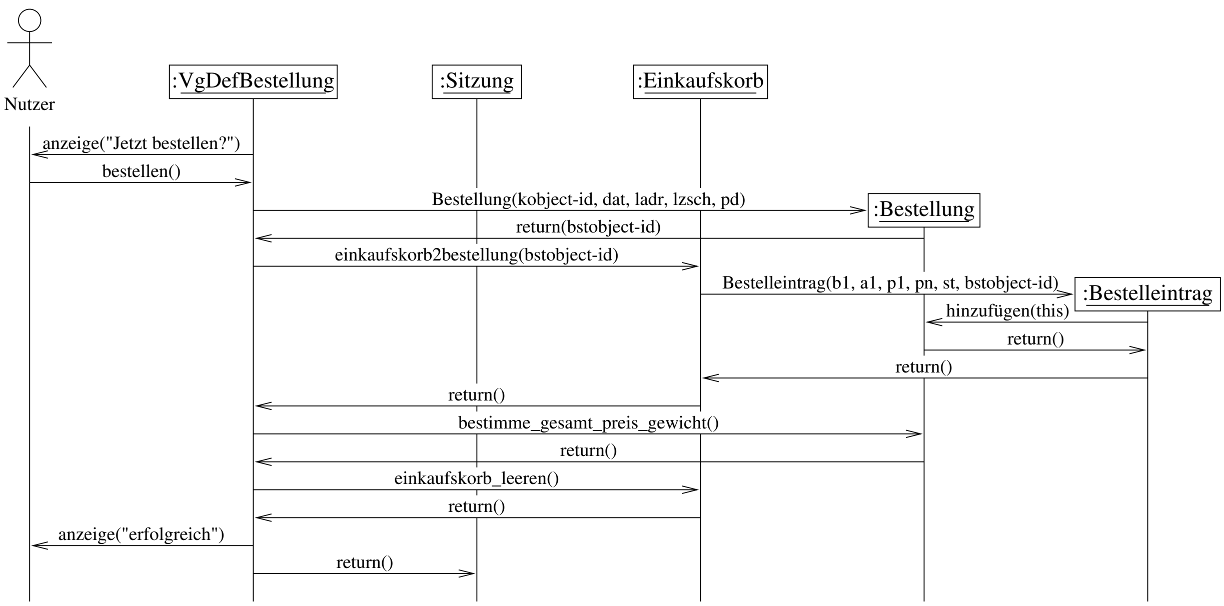


Abbildung 6.15: Sequenzdiagramm mit expliziten Objekterzeugungs–Zeitpunkten (Teil 3)

Beim dritten Teil–Sequenzdiagramm werden die im ersten und zweiten Teil erzeugten Objekte nicht referenziert und deshalb weggelassen. Auch das Paketdienstobjekt kann hier entfallen. Ein Bestellungsobjekt und ein Bestelleintragsobjekt werden explizit erzeugt, wie in Abbildung 6.15 zu sehen ist.

In Abbildung 6.16 kommt nun die oben bereits erwähnte Verfeinerungstechniken ins Spiel. Das eben erstellte dritte Teil–Sequenzdiagramm wird mit einer Schleife angereichert.

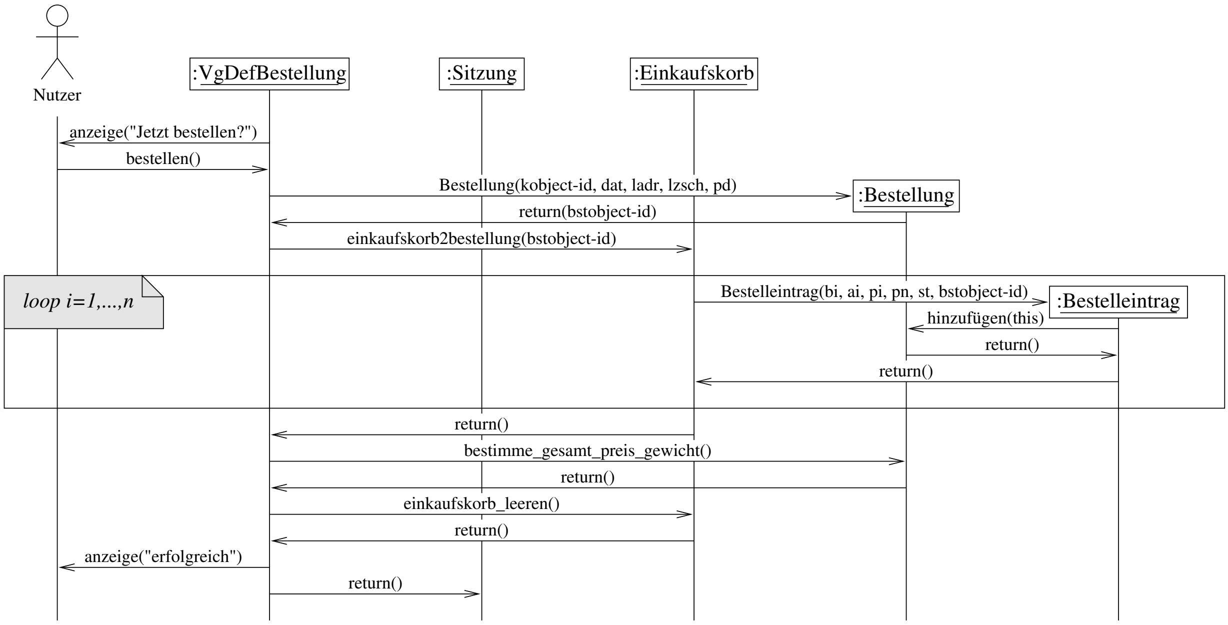


Abbildung 6.16: Sequenzdiagramm mit einer Schleife

Das neu eingefügte Rechteck umfaßt die Interaktionen, die mehrfach hintereinander ausgeführt werden müssen. Die Beschreibung im Kennzeichnungsfeld „loop i=1. . .n" kennzeichnet, daß es sich hierbei um eine Schleife handelt. Der graue Hintergrund deutet nur darauf hin, daß die Schleife (engl. loop) neu eingefügt wurde. Die Laufvariable `i` kann optional angegeben werden. Bei der Erzeugung von Bestelleinträgen werden dann die Parameter in Abhängigkeit vom Durchlauf übergeben (Variablen bi, ai, ti).

Bei den Schleifen handelt es sich um keine Beschreibungstechnik der UML. Die Darstellung wurde aus [BBH+99] übernommen. Der Schleifenzähler `i` im Kennzeichnungsfeld wurde hier neu eingeführt.

Neben Schleifen wurde vom SysLab–Team z.B. auch noch eine Darstellung für Alternativen sowie für parallele Abläufe entwickelt. Da im weiteren HMSCs für diese Darstellung verwendet werden, wird deren Anwendung nicht näher erläutert. Auch Schleifen werden in den folgenden Beispielen wieder durch HMSCs dargestellt, da sie bei dieser Arbeit besser zur Übersichtlichkeit beitragen, denn bei großen Schleifen und aufgeteilten Sequenzdiagrammen sind z.B. die Schleifen innerhalb eines Teildiagramms nicht mehr ersichtlich.

6.4.6 Highlevel MSCs

In diesem Teilabschnitt wird dieser Nutzungsfall, d.h. der zum Bestellen der im Einkaufskorb liegenden Bücher, in ein HMSC transformiert. Dabei kommen hier die HMSCs auf verschiedenen Genauigkeitsstufen zum Einsatz, d.h. sie werden geschachtelt. Zu Referenzen aus den HMSC–Knoten werden an einigen Beispielen die Teil–Sequenzdiagramme entwickelt.

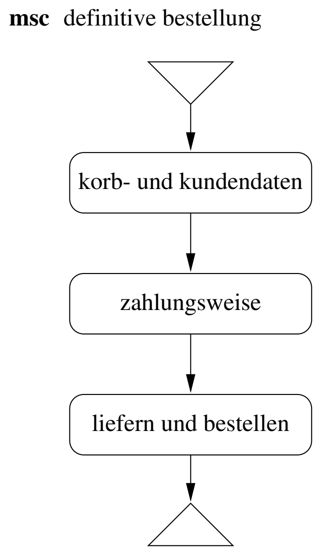


Abbildung 6.17: HMSC zur Definitiven Bestellung

Ein sehr grob–granulares HMSC zum definitiven Bestellen ist in Abbildung 6.17 dargestellt. Da HMSCs geschachtelt dargestellt werden können, d.h. die Referenzen in den Knoten können neben Sequenzdiagrammen auch auf weitere HMSCs verweisen, wurden hier nur drei Knoten gezeichnet, die alle im folgenden durch weitere HMSCs beschrieben werden.

Abbildung 6.18 zeigt das HMSC, das sich hinter dem ersten Knoten aus Abbildung 6.17 verbirgt. Hinter allen Knoten, außer der Referenz „bekannter kunde“ verbergen sich dabei Sequenzdiagramme, die in der Abbildung 6.19 dargestellt sind.

Diese Teil–Sequenzdiagramme entstanden zum einen dadurch, daß zusammengehörige Einheiten aus dem in Abschnitt 6.4.3 entwickelten exemplarischen Sequenzdiagramm übernommen wurden. Nicht involvierte Objekte konnten dabei weggelassen werden. Die Teil–Sequenzdiagramme „firmenkunde“ und „kontaktperson“ mußten neu erstellt werden. Allen Sequenzdiagrammen wurde zur Referenzierung ein Bezeichner hinzugefügt.

Bei der Eingabe der Kundendaten können bei einer wiederholten Bestellung auch die im System inhärenten Daten wieder verwendet werden. Dies verbirgt sich hinter dem HMSC–Knoten „bekannter kunde“ in Abbildung 6.18. Wie oben bereits beschrieben, referenziert dieser Knoten ein weiteres HMSC, das in Abbildung 6.20 auf Seite 105 zu sehen ist.

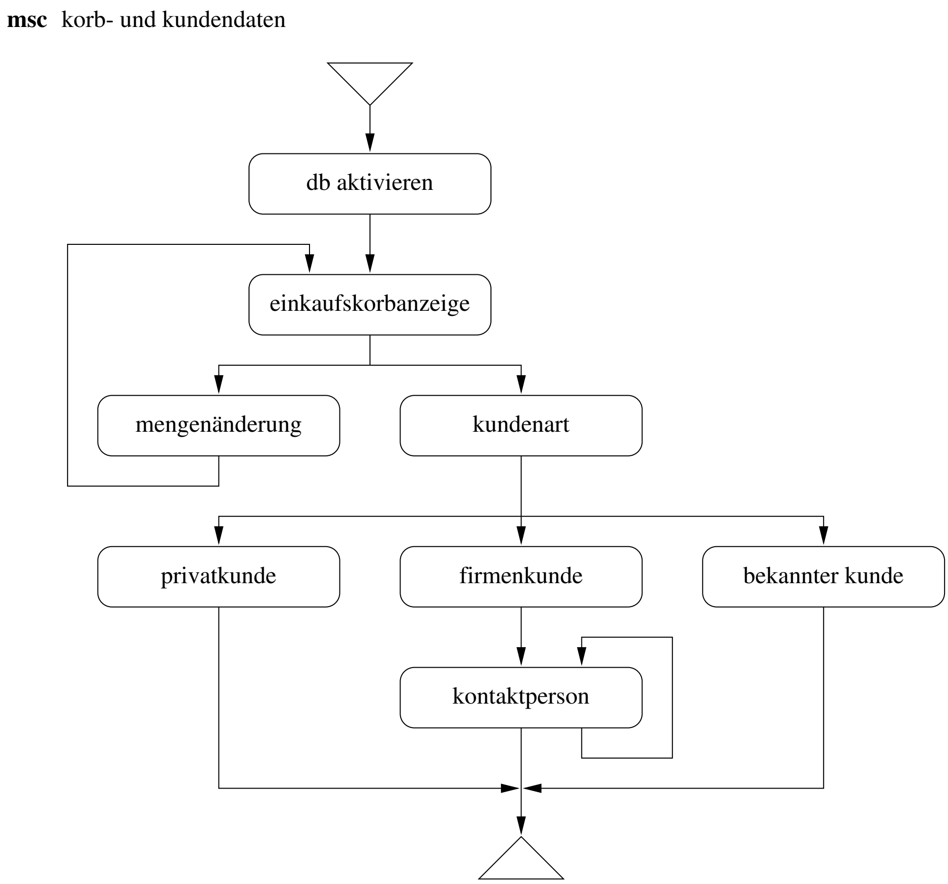


Abbildung 6.18: HMSC zur Einkaufskorbanzeige und zur Kundeneingabe

Neben einer reinen Kundenermittlung erlaubt das System auch ein Ändern der Kundendaten. Die möglichen Bearbeitungsabfolgen ergeben sich nur bei einem, dem System bekannten Kunden. Sie können aus dem HMSC zum bekannten Kunden in Abbildung 6.20 auf Seite 105 abgelesen werden. Auf eine Ausarbeitung der Teil–Sequenzdiagramme zu den Knoten des HMSCs „bekannter kunde“ wurde im Rahmen dieser Arbeit verzichtet.

Da die wiederholte Bestellung im exemplarischen Sequenzdiagramm nicht dargestellt wurde, mußte zur Ermittlung dieses HMSCs die verbale Beschreibung der objektorientierten Spezifikation aus dem Teilabschnitt 6.4.2 auf Seite 89 herangezogen werden. Hierzu waren vor allem die Variantenbeschreibungen sinnvoll. Wie dabei zu sehen ist, kann ein HMSC auch ohne die Erstellung eines Sequenzdiagramms ermittelt werden.

Die Auswahl der verschiedenen Zahlungsweisen mit deren Eingabe–Abarbeitung wird in Abbildung 6.21 auf Seite 105 als HMSC dargestellt. Eine weitere Spezifikation der Knotenreferenzen findet hier nicht statt. Beispielsweise könnte im Sequenzdiagramm zur Knotenreferenz „zahlungsweise ermitteln“ eine E–Commerce–Bezahlung ausgewählt werden. Das Sequenzdiagramm, das vom Knoten „ecommerce“ referenziert wird, müßte den Ablauf ei-

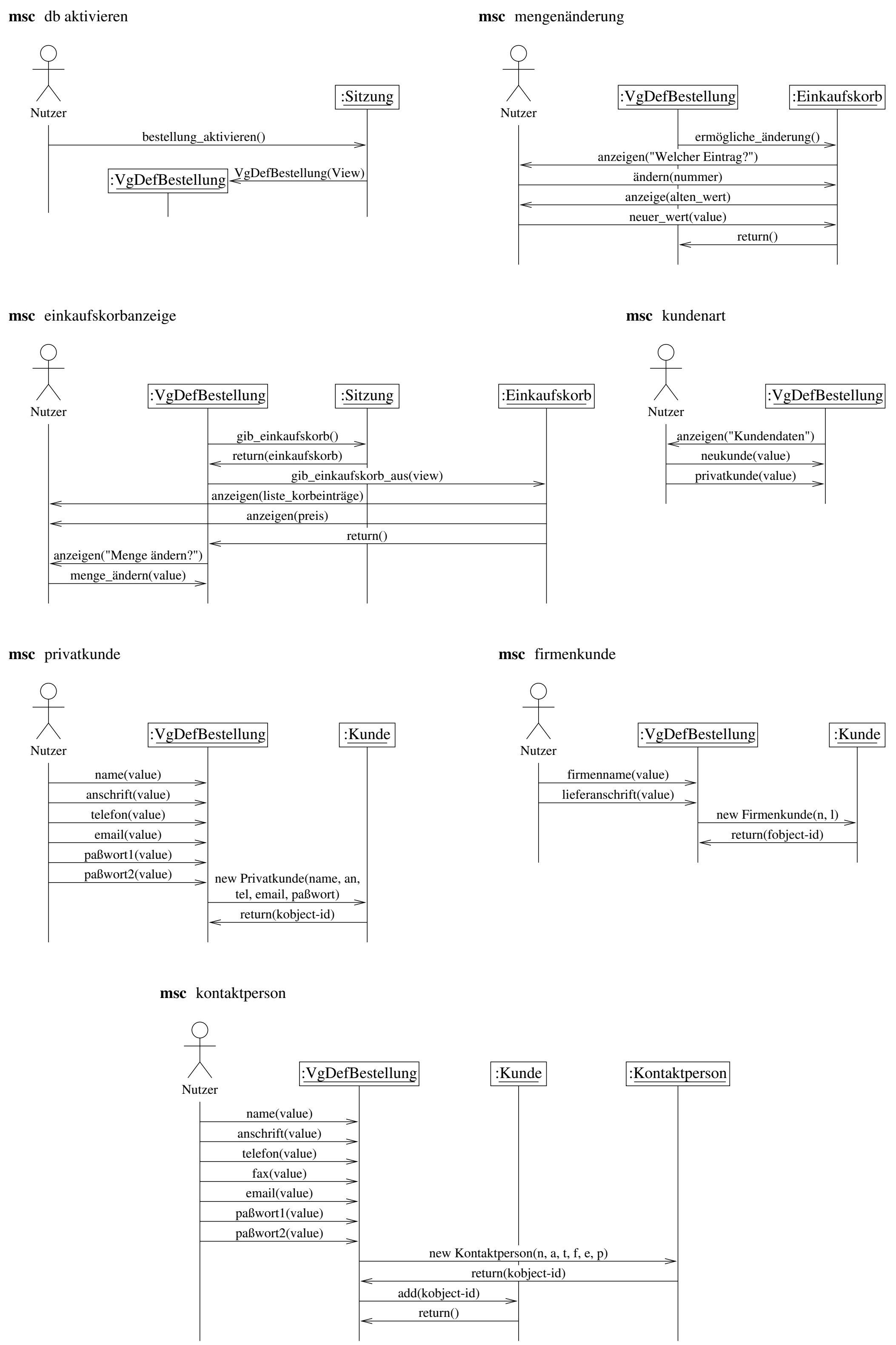


Abbildung 6.19: Teil–Sequenzdiagramme zum HMSC: korb– und kundendaten

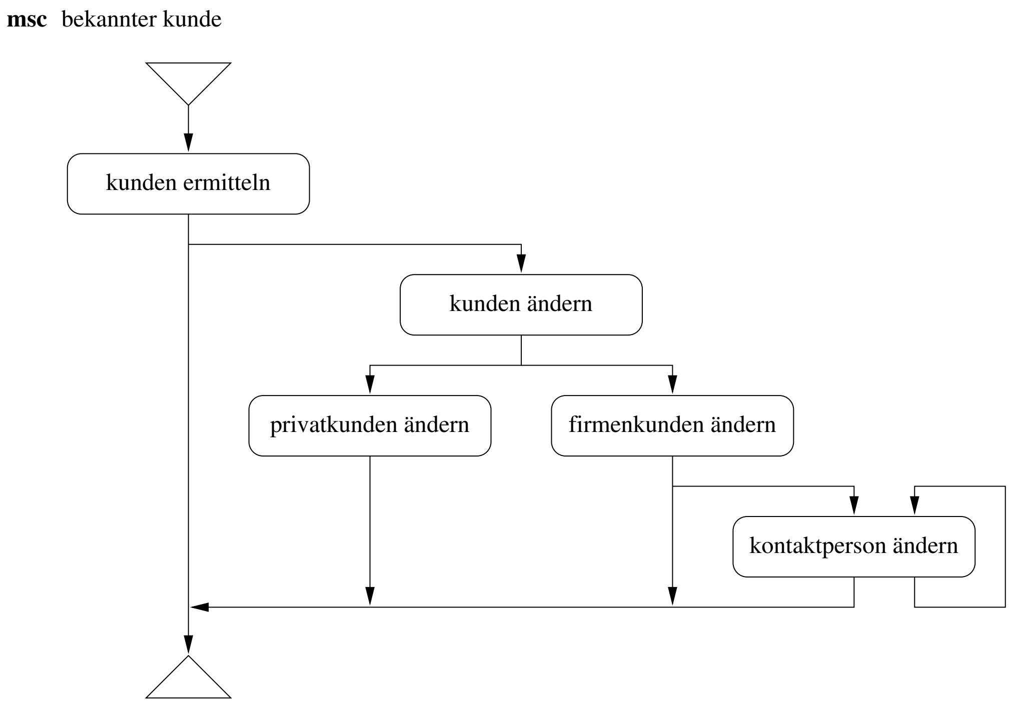


Abbildung 6.20: HMSC zur Ermittlung und Bearbeitung von Kundendaten bei einer wiederholten Bestellung

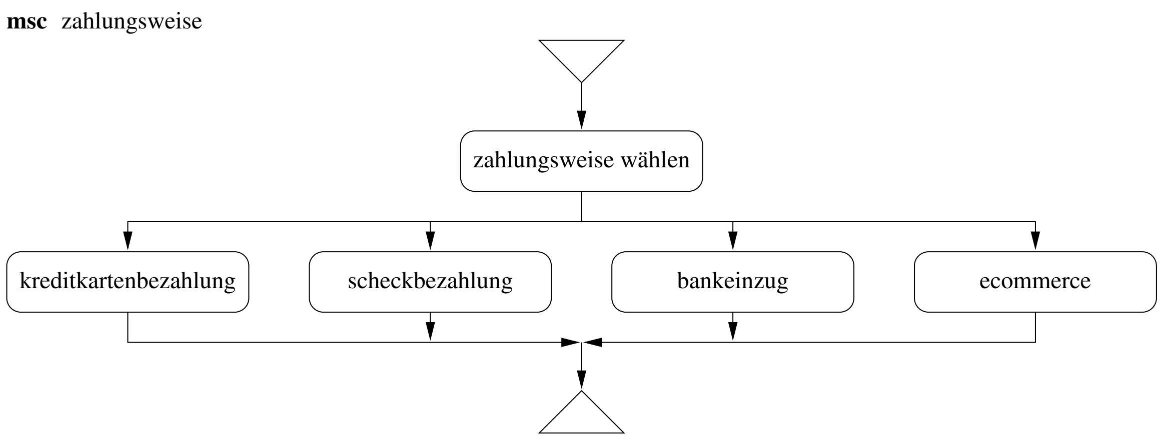


Abbildung 6.21: HMSC zur Eingabe der Zahlungsweise

ner solchen Bezahlungsweise festlegen. Bei umfangreichen Abläufen könnten auch weitere HMSCs dazwischen geschachtelt werden.

Das letzte HMSC zur definitiven Bestellung behandelt noch die Paketdienstauswahl, die optionale Angabe einer speziellen Lieferanschrift und die endgültige Bestellungsaufgabe (siehe Abbildung 6.22 auf der nächsten Seite). Auch hier wurden die Knotenreferenzen nicht weiter bearbeitet.

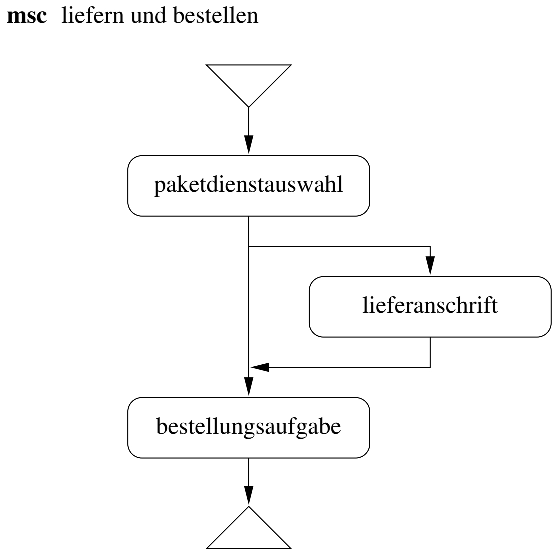


Abbildung 6.22: HMSC zur Eingabe von Lieferdaten und zur Bestellungsaufgabe

6.4.7 Parallelisierung der Bestellungsabarbeitung

Die letzte Analyse im Nutzungsfall zur definitiven Bestellung beschäftigt sich mit einer möglichen Parallelisierung des Bestellvorgangs. Das exemplarische Sequenzdiagramm aus Abbildung 6.16 auf Seite 101 wird dementsprechend modifiziert und das HMSC aus Abbildung 6.22 wird ergänzt.

Sobald ein Kunde alle seine Bestelldaten eingegeben hat und die Bestellung bestätigt wurde, kann das System in der bereits gezeigten Weise die Bestelldaten aufnehmen und der Benutzer kann sich bereits zu diesem Zeitpunkt z.B. weiteren Suchanfragen widmen. In diesem Stadium ist eine Parallelisierung möglich. Abbildung 6.24 auf Seite 108 zeigt die nötigen Veränderungen am exemplarischen Sequenzdiagramm.

Da es bis jetzt nicht notwendig erschien, wurden bei den bis jetzt dargestellten Sequenzdiagrammen keine Aussagen bezüglich der Aufrufart getroffen. Die Spitzen der Nachrichtenpfeile in den Sequenzdiagrammen wurden deshalb immer offen gezeichnet. Eine solche Darstellung (dargestellt durch offene Pfeilspitzen) läßt es offen, wie später die Nachrichtenkommunikation stattzufinden hat. Bei der weiteren Bearbeitung dieses Teilabschnitts wird eine Unterscheidung zwischen synchronen und asynchronen Nachrichten getroffen. Die Unterscheidung wird im Diagramm durch eine unterschiedliche Art der Pfeilspitzendarstellung präsentiert.

Die Nachricht, die die parallele Bestellungsabarbeitung im System einleitet, wird im Diagramm mit einer halbierten, ausgemalten Pfeilspitze dargestellt. Dieses Symbol wird in Anlehnung an [BBH+99] für asynchrone Nachrichten verwendet. Beim Beenden der Bestellungsabarbeitung wird vom System an den Nutzer nochmals eine asynchrone Nachricht

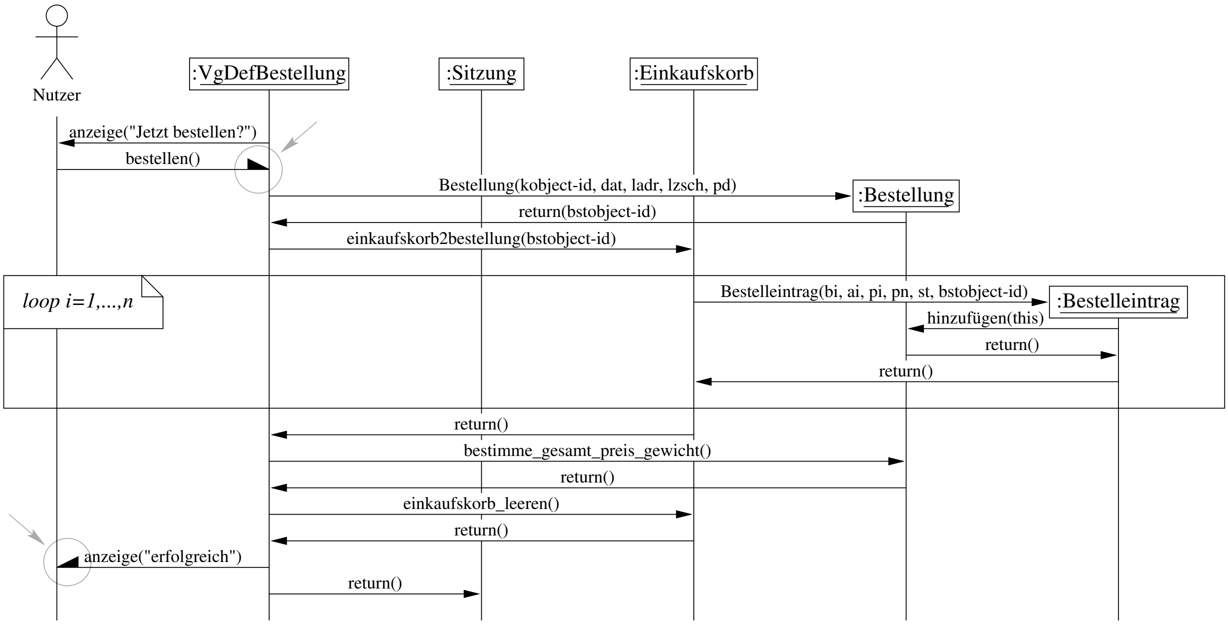


Abbildung 6.23: Sequenzdiagramm zur definitiven Bestellung mit Unterscheidung zwischen synchronen und asynchronen Nachrichten

gesendet, die über das Bearbeitungsende informiert.

Die restlichen Nachrichten sind synchrone Nachrichten, d.h. jede aufrufende Funktion blockiert solange, bis die aufgerufene Funktion die Kontrolle an sie zurückgibt[2]. Die synchronen Nachrichten werden durch ganze, ausgefüllte Pfeilspitzen dargestellt.

In den Sequenzdiagrammen könnten auch noch die aktiven Phasen der Objekte eingezeichnet werden (vgl. „Activations“ in [BBH$^+$99]). Da sie hier nicht unbedingt erforderlich sind, denn die Objektaktivitäten können auch über die ein– und ausgehenden Nachrichten festgestellt werden, und das Diagramm weiter mit Information füllen würden, so daß es an Übersichtlichkeit verlieren würde, wurden die „Activations“ nicht verwendet.

In Abbildung 6.24 wurde die Parallelisierung gemäß Abbildung 6.23 in das HMSC aufgenommen. Die gewählte Darstellung von parallelen Abläufen in Highlevel Message Sequence Charts entspricht dem Standard MSC–96 und wurde aus [Krü99] übernommen. Die in der Box eingezeichneten Teil–HMSCs mit eigenem Start– und Endesymbol zeigen zwei unabhängige Vorgänge, die parallel bearbeitet werden können.

Der parallele Ablauf unterliegt jedoch einer Beschränkung, die hier aus dem HMSC nicht ersichtlich ist. Sobald die Kundensitzung wieder auf den Einkaufskorb zugreifen will, um z.B. Einträge darin zu vermerken, muß die Bestellabarbeitung beendet sein.

[2]In [BBH$^+$99] werden synchrone Nachrichten als „Calls“ und asynchrone Nachrichten als „Messages“ bezeichnet.

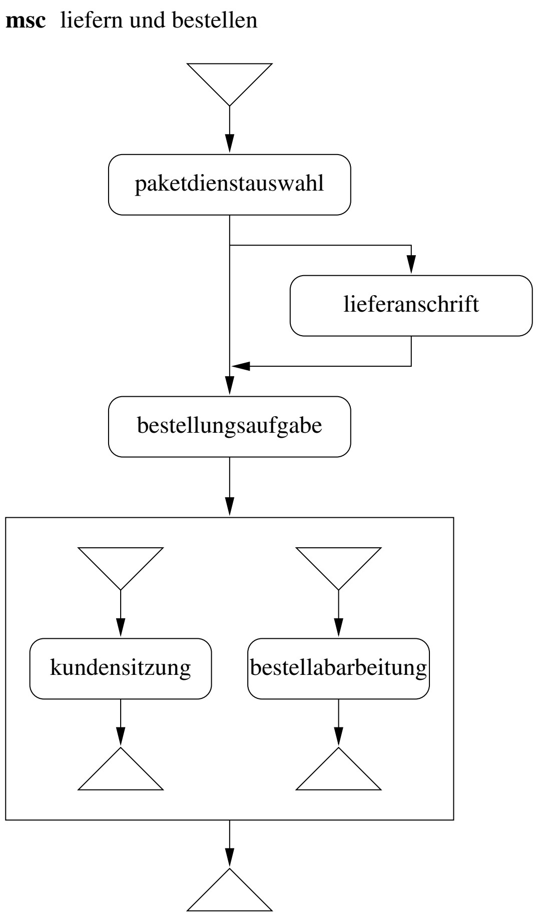


Abbildung 6.24: Ergänzung des HMSC zur parallelen Abarbeitung der Bestellungsübernahme

6.4.8 Nachbemerkung

Bei diesem Nutzungsfall wurden auch wieder ausgehend von der Produktfunktionen–Beschreibung im Pflichtenheft und vom Systemszenario zur definitiven Bestellung eine objektorientierte Spezifikation mit Nutzungsfalldiagramm, ein exemplarisches Sequenzdiagramm mit Ergänzungen und verschiedene HMSCs erstellt.

Außerdem wurden Teil–Sequenzdiagramme für HMSC–Referenzen erstellt, Verfeinerungen in Sequenzdiagrammen bezüglich Objekterzeugung, Schleifendurchläufen und Kommunikationsart (synchron, asynchron) angewendet und parallele Abläufe sowohl in Sequenzdiagrammen sowie in HMSCs betrachtet.

Das Ablaufverhalten des Nutzungsfalls, d.h. sein dynamisches Verhalten, wurde auf diese Art genauer analysiert und spezifiziert. Eine abschließende Diskussion der verwendeten Techniken findet in der Zusammenfassung zu diesem Kapitel statt, siehe Abschnitt 6.6 auf Seite 115.

6.5 Büchersuche

Der Kunde will ein bestimmtes Buch oder eine Reihe von Büchern im Online–Katalog finden. Eine Autoren– und eine Titelsuche wird ebenso wie eine Suche nach einem Interessensgebiet vom System angeboten. Neben diesen potentionellen Mehrfachselektionen existiert noch die Auswahl mittels ISBN–Nummer, die das Vorhandensein des zugehörigen Artikels im Katalog prüft und gegebenenfalls die gewünschten Informationen preisgibt.

Der Nutzungsfall Büchersuche wird in diesem Abschnitt ausgehend von der Pflichtenheftfunktion und vom Systemszenario (Teilabschnitt 6.5.1) genauer spezifiziert. Nach der objektorientierten Spezifikation (Teilabschnitt 6.5.2) zur genauen Analyse des Nutzungsfalls wird ein exemplarisches Sequenzdiagramm erzeugt (Teilabschnitt 6.5.3). Alternativen hierzu werden nur informell erfaßt. An dem erzeugten Sequenzdiagramm wird zum Abschluß im Teilabschnitt 6.5.4 noch eine Objektverfeinerung durchgeführt.

6.5.1 Vorausgehende Spezifikation

Die Produktfunktion */F:Suche/* aus Seite 12 im Pflichtenheft und das Systemszenario zur Büchersuche, welches im Abschnitt 5.2 auf Seite 56 entwickelt wurde und in Abbildung 6.25 nochmals gezeigt wird, bilden die Basis für die Ermittlung des Szenarios Büchersuche auf der Ebene der Nutzungsfälle.

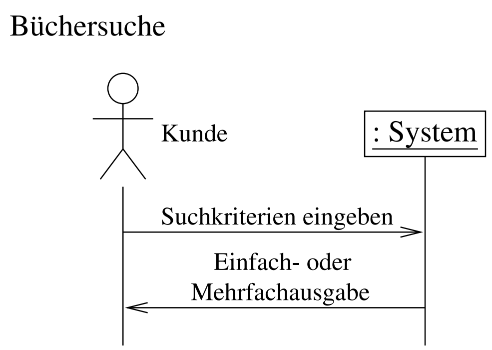


Abbildung 6.25: Systemszenario: Büchersuche

Eine Verfeinerung des Nachrichtenflusses zwischen Nutzer und System und eine Objektverfeinerung bezüglich der Fachklassen und eines Vorgangsobjekts mit seinen Auswirkungen findet im folgenden statt.

6.5.2 Objektorientierte Spezifikation

Vor der Erstellung eines exemplarischen Sequenzdiagramms muß noch der Nutzungsfall unter objektorientierten Gesichtspunkten spezifiziert werden.

Eingabedaten

Die Suchbegriffe sind hier die Eingabedaten. Es können Autor, Titel und Interessengebiet einzeln oder gemischt eingegeben werden. Bei der ISBN–Nummer reicht diese allein aus, denn eine Abbildung hiervon ist bei gültigen Nummern immer injektiv, d.h. verschiedene Bücher mit gleichen ISBN–Nummern existieren nicht. Natürlich existieren mehrere Exemplare eines Buches mit selbstverständlich gleicher ISBN–Nummer.

Ausgabedaten

Falls Bücher gefunden werden, die den Suchbegriffen entsprechen, wird dem Benutzer eine Liste angezeigt. Die Einträge in der Liste enthalten den Autor, den Titel des Buches und den Preis. Für genauere Informationen enthalten sie einen Verweis auf das Buch im Buchkatalog. Falls keine der Anfrage entsprechenden Bücher gefunden wurden, erscheint eine Benachrichtigung.

Diese Ausgabeliste und eine interne Liste der Suchergebnisse für die weitere Bearbeitung, z.B. zum Vormerken für die Bestellung, stellen die Ausgabedaten des Nutzungsfalls dar.

Akteur

Als Akteur tritt hier wieder der Käufer auf. Er gibt die Stichwörter ein und das System präsentiert ihm daraufhin die gewünschten Informationen.

Veränderte Fachklassen

Die Fachklasse Sitzung wird verändert. Die Ergebnisliste muß hier für weitere Bearbeitungen zwischengespeichert werden.

Beschreibung

Zu Beginn einer Büchersuche gibt der Bookshop–Nutzer seine Suchkriterien in ein gemeinsames Datenfeld ein. Autor, Titel, Interessengebiet und/oder ISBN–Nummer können ohne Einhaltung einer Reihenfolge eingegeben werden. Auch mehrere Autoren, Titel und Interessen können in einer Anfrage angegeben werden.

Die Suche wird dann durch Drücken der Return– oder der Maus–Taste gestartet.

Nachdem die Suchanfrage im System bearbeitet wurde, wird dem Kunden das Ergebnis bei erfolgreicher Suche in Form einer Liste präsentiert, die entweder nur einen Eintrag oder mehrere enthalten kann. Diese Liste enthält nur eine Übersichtsdarstellung der gefundenen Artikel. Will der Kunde genauere Informationen zu einem Buch, wird der Listeneintrag selektiert und eine Beschreibungsseite mit den kompletten Buchdaten angezeigt.

Eine Übereinstimmung der Suchbegriffe mit den Inhalten in der Datenbank wird folgendermaßen abgearbeitet: Zuerst werden diejenigen Bücher angezeigt, die mit allen eingegeben Kriterien übereinstimmen. Liegen keine Einträge vor, die allen Suchkriterien gemeinsam entsprechen, so werden zu jedem einzelnen Kriterium die gefundenen Bücher ausgegeben.

Wurden bei einer Suchanfrage keine Treffer erzielt, so erfolgt nur eine Benutzerinformation mit Wiederholung der Suchanfrage in der Eingabemaske. Der Benutzer kann dadurch die Eingabe modifizieren und damit Suchkriterien abändern oder Eingabefehler ausbessern.

Variante 1

Einfache Expertensuche. Die Suchbegriffe werden vom Anwender nicht in eine gemeinsame Maske eingetragen, sondern es existieren spezielle Eingabefelder für Autor, Titel, Interessengebiet und ISBN–Nummer. Hierbei können reguläre Ausdrücke verwendet werden.

Variante 2

Komplexe Expertensuche. Der Anwender kann seine Suchanfrage strukturiert in einem SQL[3]–Statement eingeben. Das System unterstützt hierbei auch komplexe Suchanfragen.

Graphische Darstellung

Das Nutzungsfalldiagramm in Abbildung 6.26 beschreibt wieder den Zusammenhang zu anderen Nutzungsfällen im System.

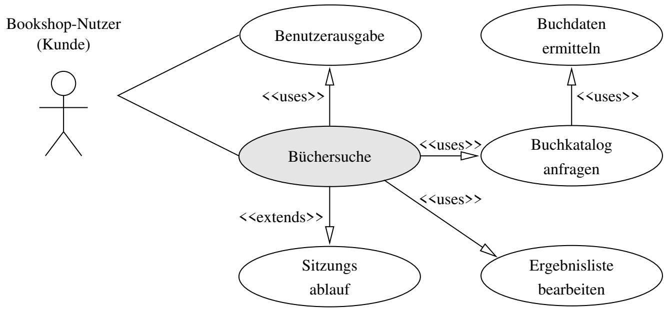


Abbildung 6.26: Nutzungsfalldiagramm: Büchersuche

Der Nutzungsfall „Büchersuche“ verwendet die Nutzungsfälle „Benutzerausgabe“, „Buchkatalog anfragen“ und „Ergebnisliste bearbeiten“. „Buchkatalog anfragen“ verwendet seinerseits noch „Buchdaten ermitteln“. Bei diesen Verwendungs–Relationen handelt es sich um USES–Beziehungen.

Die Liste der gefundenen Bücher wird für eine spätere Weiterverwendung von der Sitzung verwaltet werden. Der Nutzugsfall „Sitzungsablauf“ muß entsprechend erweitert werden. Bei der Erweiterung handelt es sich um eine EXTENDS–Beziehung.

6.5.3 Exemplarische Verhaltensspezifikation

Das Sequenzdiagramm (Abbildung 6.27) zeigt ein Beispiel für den Nutzungsfall, bei dem der Kunde den Autor, den Buchtitel und den Sachbereich in einer eigenen Maske angibt

[3]Structured Query Language. Hierbei handelt es sich um eine weit verbreitete Anfragesprache, die vor allem bei relationalen Datenbanken Anwendung findet.

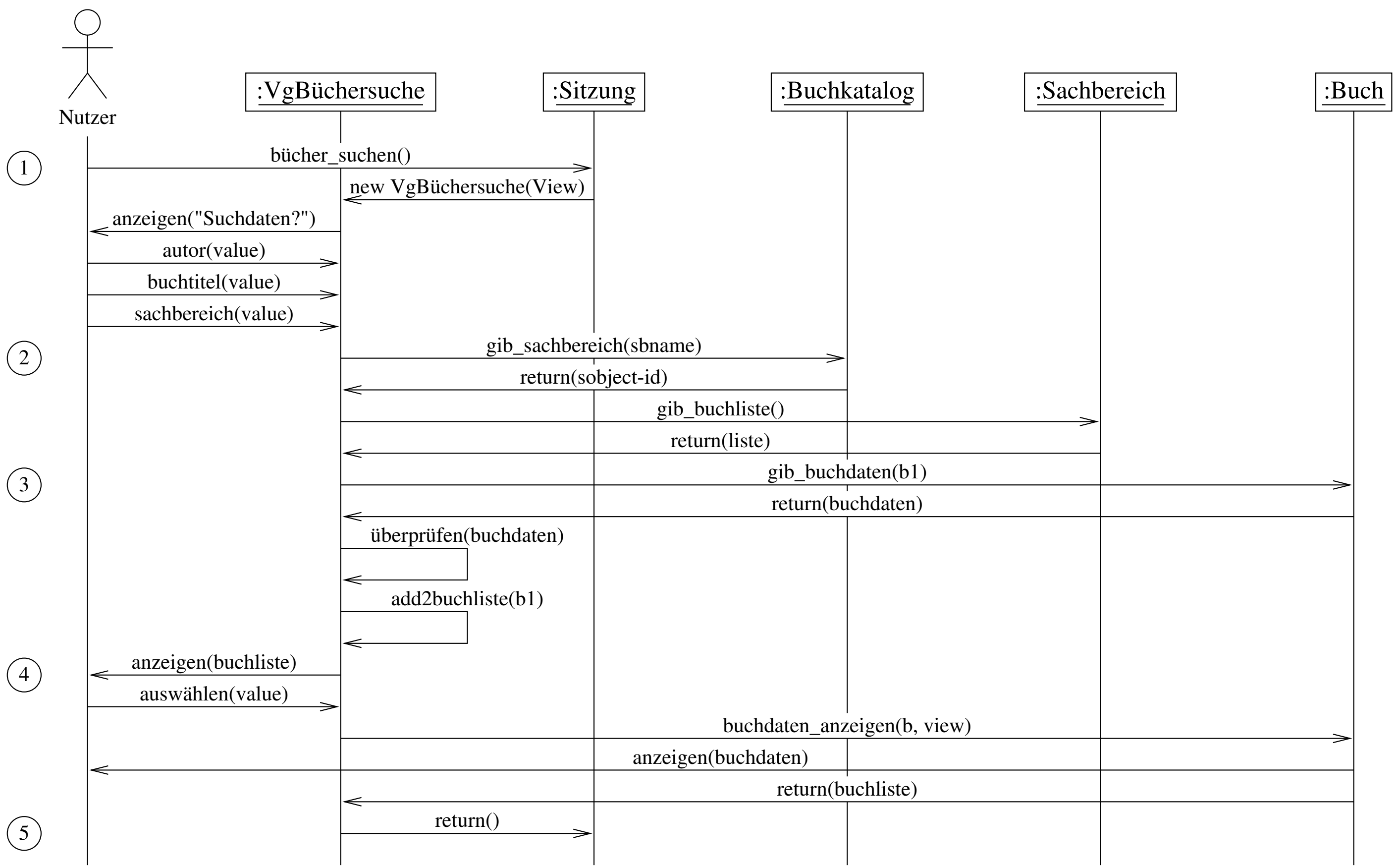


Abbildung 6.27: Sequenzdiagramm zum Nutzungsfall: Büchersuche

(dies entspricht der Variante 1 aus der objektorientierten Spezifikation). Die Suche wird daraufhin vom System gestartet und die Ergebnisliste dem Anwender präsentiert. Die Ergebnisliste wird am Ende an das System zur Weiterverwendung übergeben.

Die verwendeten Fachobjekte leiten sich aus der Struktur des Anwendungskerns (Kapitel 4) ab. Das neu hinzugefügte Objekt „VgBüchersuche“ steuert als Vorgangsobjekt den Ablauf. Folgende Anmerkungen sind zum exemplarischen Sequenzdiagramm noch anzufügen:

Punkt 1: Das Sitzungsobjekt erzeugt wie bei allen anderen Nutzungsfällen zur Kundensitzung das Vorgangsobjekt zur Büchersuche.

Die Daten werden hier getrennt in die Dialogfelder für Autor, Buchtitel und Sachbereich eingegeben. Eine Suchanfrage, bei der alle Daten gemeinsam in einem einzigen Feld eingegeben werden, kann vom System ebenfalls bearbeitet werden.

Punkt 2: Die Bucheinträge im Buchkatalog zum angegebenen Sachbereich werden ermittelt. Alternativ müßte bei einer unsortierten Eingabe der ganze Buchbestand durchsucht werden.

Punkt 3: Die Buchdaten jedes Buches zum Sachbereich müssen aus dem Katalog geholt werden und mit den Sucheingaben verglichen werden. Dieser Vorgang wurde im Sequenzdiagramm nur exemplarisch für ein Buch dargestellt.

Entsprechen die Buchdaten denen der Suche, erfolgt eine Aufnahme des Buches in die Ergebnisliste.

Punkt 4: Die Ergebnisliste wird dem Anwender präsentiert. Er wählt einen Eintrag aus und bekommt hierzu genauere Daten aus dem Buchkatalog.

Alternativ können keine oder mehrere Einträge ausgewählt und betrachtet werden.

Punkt 5: Die Ergebnisliste wird an das Sitzungsobjekt zurückgegeben, da sie z.B. zum Vormerken im Einkaufskorb weiter verwendet wird.

6.5.4 Objektverfeinerung

An dem eben erstellten Sequenzdiagramm wird eine Objektverfeinerung angewendet, wie sie in [BBH+99] aufgeführt ist. Ein Objekt spaltet sich hierbei in weitere Einzelobjekte auf. In diesem Beispiel bleibt das ursprüngliche Objekt bestehen, es tritt lediglich einige seiner Aufgaben an die neu eingefügten Objekte ab. In Abbildung 6.28 auf der nächsten Seite ist die Objekterweiterung zu sehen. Die ergänzten Objekte mit ihren Interaktionen sind zur Verdeutlichung grau hinterlegt.

Das Vorgangsobjekt zur Büchersuche spricht nun ein eigenes Suchobjekt an, welches den Suchvorgang steuert. Wird z.B. an Stelle der hier verwendeten „Zusammengesetzten Suche“ eine „Expertensuche“ verwendet, so kann hier das entsprechende Suchobjekt erzeugt

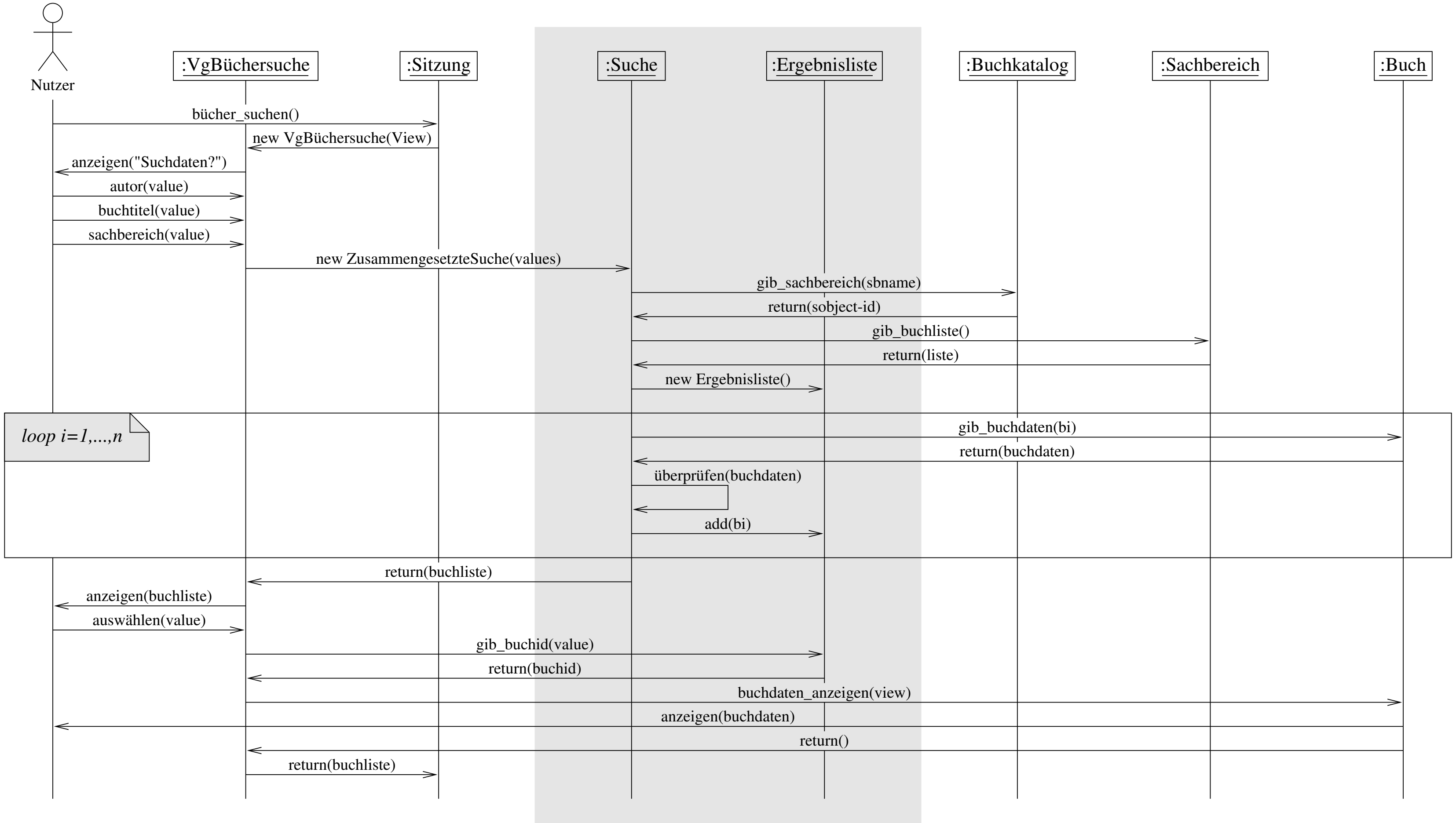


Abbildung 6.28: Erweitertes Sequenzdiagramm zum Nutzungsfall Büchersuche

werden. Die Auswirkungen auf die Klassenstruktur werden hierzu im Abschnitt 8.5 auf Seite 142 erläutert.

Die Ergebnisliste wird ebenfalls als eigenes Objekt dargestellt. Dieses kann später an das Sitzungsobjekt aus den oben genannten Gründen weitergegeben werden.

Da sich die Suche über mehrere Bücher aus dem Buchkatalog erstreckt, wurde hierzu noch eine Schleife eingefügt. Eine Diskussion der Schleifendarstellung erfolgte bereits in Teilabschnitt 6.4.5 auf Seite 98.

6.5.5 Nachbemerkung

Eine Objektverfeinerung wurde anhand der Büchersuche in einem Sequenzdiagramm durchgeführt. Hierzu war wieder eine vorausgehende objektorientierte Spezifikation und die Erstellung des exemplarischen Sequenzdiagramms notwendig. Ausgangspunkt hierfür waren, wie in den Nutzungsfällen zuvor, die Produktfunktion aus dem Pflichtenheft und das Szenario auf der Ebene des Systems zur Büchersuche.

Bei der Objektverfeinerung wurden Teilaufgaben an die neuen Objekte abgetreten. Dementsprechend mußten die Interaktionen teilweise zu den neuen Objekten geführt werden. Außerdem kamen auf diese Art neue Interaktionen zwischen den neu eingeführten Objekten zum Vorschein. Ein neu erstelltes Objekt kann z.B. nach Beendigung des Vorgangsobjekts weiter existieren und z.B. zur Datenweitergabe im System verwendet werden.

Neben der Verfeinerung der Objekte wurde auch eine Schleifendarstellung im Sequenzdiagramm nochmals dargestellt.

6.6 Zusammenfassung

Szenarien auf Nutzungsfallebene wurden ausgehend von den Produktfunktionen im Pflichtenheft und den in Kapitel 5 entwickelten Systemszenarien in diesem Kapitel erstellt. Dazu wurden objektorientierte Spezifikationen, exemplarische Sequenzdiagramme und Highlevel Message Sequence Charts verwendet. An den Sequenzdiagrammen wurden Darstellungsalternativen und Verfeinerungen gezeigt, die vom rein exemplarischen Verhaltensablauf hin zu einer vollständigen Verhaltensbeschreibung der Objekte führten. Mit Hilfe von Highlevel Message Sequence Charts konnte dieses allgemeine Verhalten ebenfalls dargestellt werden, wobei diese besonders zur Darstellung von Alternativabläufen sehr geeignet sind. Außerdem wurde mit Hilfe der Sequenzdiagramme und der HMSCs auch noch ein möglicher paralleler Ablauf spezifiziert.

Da die Nutzungsfälle auf Systemebene im vorausgehenden Kapitel für die Ausarbeitung in diesem Kapitel noch zu ungenau beschrieben waren, stand normalerweise am Anfang vor den weiteren Bearbeitungsschritten die objektorientierte Spezifikation. Eine Ausnahme bildet dabei die Suche, die nur ein übergeordneter Nutzungsfall zum „Blättern im Katalog" und zur „Büchersuche" ist. Bei der objektorientierten Spezifikation wurden der Nutzer, die Ein– und Ausgabedaten, die veränderten Fachklassen, eine Beschreibung des Nutzungsfallverhaltens und mögliche Alternativverhalten genau erfaßt und beschrieben. Die Zusammenhänge mit anderen Nutzungsfällen und die Einwirkung des Nutzers wurden in Nutzungsfalldiagrammen graphisch dargestellt. Im Gegensatz zu Sequenzdiagrammen, die das Ablaufverhalten eines Nutzungsfalls darlegen, zeigen Nutzungsfalldiagramme die Anwendung anderer Nutzungsfälle. Die Nutzungsfälle erwarten, daß ein Teil des gewünschten Verhaltens von einem anderen Nutzungsfall erledigt wird. Stellt eine anderer Nutzungsfall hierzu nichts bereit, muß er erweitert werden.

Zu den Ein– und Ausgabedaten kamen auch die zwischen den Nutzungsfällen im System ausgetauschten Daten hinzu. Obwohl zu Beginn der Bearbeitung die Aufnahme von systeminternen Daten zu den Ein- und Ausgaben unklar erschien, stellte sich während der Bearbeitung deren Hinzufügen als vorteilhaft heraus, denn die systeminternen Daten werden bei der objektorientierten Spezifikation an keiner anderen Stelle erwähnt. Wäre ihre Spezifikation bei den Ein- und Ausgabedaten nicht erfolgt, hätten sie z.B. bei der Erstellung von Sequenzdiagrammen genau analysiert werden müssen.

Die objektorientierte Spezifikation stellt neben dem Zusammenhang zum Pflichtenheft und zum Systemszenario, die als Ausgangspunkte dienten, auch einen Zusammenhang zu den Fachklassen her. Die veränderten Fachklassen werden herausgearbeitet und somit beginnt hier bereits eine Vereinigung der Nutzungsfälle mit den Fachklassen.

Die verbalen Beschreibungen des Nutzungsfallablaufs spezifizieren das dynamische Verhalten. Die Variantenbeschreibungen legen mögliche Alternativabläufe fest.

Nutzungsfalldiagramme zeigen vor allem den Zusammenhang zu anderen Nutzungsfällen. Hierbei können reine Verwendungsbeziehungen sowie notwendige Ergänzungen von bestehenden Nutzungsfällen in Erweiterungsbeziehungen dargestellt werden. Da die Intention von Nutzungsfalldiagrammen variieren kann, werden sie z.B. in [Bre98a] ganz außer acht gelassen. Da sie aber in dieser Arbeit, wie oben erwähnt, zur Darstellung von Nutzungsfallbeziehungen verwendet wurden, erwiesen sie sich als sinnvoll und wurden zu diesem Zweck auch eingesetzt.

Aus der objektorientierten Spezifikation wurden in erster Linie exemplarische Sequenzdiagramme erstellt. Aus der Verhaltensbeschreibung konnten sogenannte „Sunny–Day–Sequenzdiagramme" erstellt werden, die den planmäßigen Ablauf des Nutzungsfalls beispielhaft darstellen. Zu den Varianten oder zu außerplanmäßigen Verhaltensabläufen wären ebenfalls eigene exemplarische Sequenzdiagramme denkbar. Da dies aber eine Vielzahl von Sequenzdiagrammen hervorrufen würde und auch nicht jede Alternative auf diese Art und Weise dargestellt werden muß, wurden hier keine sogenannten „Rainy–Day–Sequenzdiagramme" dargestellt. Die Darstellung von Alternativ– und Schleifenabläufen erledigen in dieser Arbeit Highlevel Message Sequence Charts, die sich sehr schön für eine vollständige Beschreibung des Interaktionsverhaltens auf dieser Ebene eignen. Sequenzdiagramme eignen sich sehr schön zur Darstellung von internen Nachrichtenflüssen zwischen einem Vorgangsobjekt und Fachobjekten. Das Vorgangsobjekt steuert den Verhaltensablauf und ist Bindeglied zwischen Nutzer und den Fachobjekten. Die Fachobjekte sind Instanzen der Fachklassen und sie stehen neben den Vorgangsobjekten an Stelle der Systemobjekte in den Systemszenarien. Sequenzdiagramme können auch noch weiter verfeinert werden. Erstens können die Objekte oftmals noch in weitere Einzelobjekte aufgeteilt und zweitens die Nachrichtenflüsse genauer spezifiziert werden. Die Verfeinerung der Nachrichtenflüsse konnte beim Übergang vom Systemszenario zu dem auf Nutzungsfallebene betrachtet werden. Die Objektverfeinerung wurde an einem Beispiel gezeigt.

Sequenzdiagramme können z.B. auch mit Schleifen verfeinert werden. Wiederholte Abläufe können dadurch zusammengefaßt oder exemplarische Abläufe vollständig dargestellt werden. Auch Alternativabläufe können in Sequenzdiagrammen gezeigt werden. Da aber in dieser Arbeit HMSCs eingesetzt wurden, kamen zur Alternativdarstellung und zur Schleifendarstellung diese zum Einsatz. Sie sind übersichtlicher, da sie keine Lebenslinien der Objekte enthalten und die Alternativabläufe können z.B. in einem Knoten referenziert werden. Für diese übersichtliche Darstellung erlauben HMSCs außerdem eine Schachtelung, wie bei deren Anwendung beschrieben wurde. Bei Schleifen mit vielen inhärenten Interaktionen sind HMSCs viel übersichtlicher. Auch die Erarbeitung von Alternativen und Schleifen erweist sich mit HMSCs leichter. Sie wurden in dieser Arbeit den erweiterten Sequenzdiagrammen aus den aufgeführten Gründen vorgezogen. Als Nachteil bei den HMSCs erweist sich, daß der Ablauf oftmals in sehr vielen einzelnen Diagrammen dargestellt wird.

Als Erweiterung für HMSCs wäre eine Ergänzung von Bedingungen an den Kanten des Graphen sinnvoll. Dadurch könnte die Aussagekraft von HMSCs weiter gesteigert werden, so daß z.B. die Wahl des eingeschlagenen Weges ohne Betrachtung weiterer HMSCs oder Sequenzdiagramme nachvollzogen werden könnte. Die Terminierung von Schleifendurchläufen könnte dadurch ebenfalls dargestellt werden.

Parallele Abläufe können sowohl mit HMSCs als auch mit Sequenzdiagrammen dargestellt werden. Bei HMSCs führt dies zu einer neuen Darstellung, während hierzu bei Sequenzdiagrammen nur die Pfeilspitzen unterschiedlich gezeichnet werden. Vor– und Nachteile der verschiedenen Diagrammtypen kamen bei der Anwendung nicht zum Vorschein.

Zwischen der vorausgehenden Spezifikation und der objektorientierten Spezifikation war die Erarbeitung von neuer Information erforderlich, denn die vorausgehenden Spezifikationen reichten hierzu allein nicht aus. Beim Übergang auf exemplarische Sequenzdiagramme war der größte Teil der Information bereits aus der objektorientierten Spezifikation zu entnehmen, ebenso beim Übergang zu den HMSCs. Schleifen und Alternativen konnten aus den Varianten in der objektorientierten Spezifikation übernommen werden. Die Trennung zur Parallelisierung wurde im Sequenzdiagramm ersichtlich. Informationen dafür waren in vorausgehenden Kapiteln nicht enthalten, sie mußten hier erarbeitet werden.

Szenarien auf Systemebene eignen sich sehr gut, das dynamische Verhalten zu erfassen und zu beschreiben. Die verwendeten Diagrammtypen, Nutzungsfalldiagramme, Sequenzdiagramme und HMSCs, eignen sich sehr gut, das Verhalten und den Zusammenhang zu anderen Nutzungfällen und Objekten herauszufinden. Je nach Umfang des Nutzungsfalls kann die Genauigkeit der Analyse und der Darstellung sowohl in Nutzungsfalldiagrammen als auch in Sequenzdiagrammen und HMSCs variieren.

Das Verhalten der Nutzungsfälle konnte mit Hilfe der Sequenzdiagramme und HMSCs soweit vollständig beschrieben werden. Eine genauere Betrachtung, bei der die z.B. bis jetzt noch nicht erwähnten Programmabbrüche durch den Anwender ergänzt werden, findet im nächsten Kapitel bei der Betrachtung des Objektverhaltens zu Einzelobjekten statt. Sicherlich könnten solche Alternativabläufe auch hier bereits eingefügt werden, aber da in den Diagrammen oftmals eine Vielzahl von Objekten betrachtet werden und für jedes dieser Objekte alle notwendigen Alternativen eingezeichnet werden müßten, wäre an eine übersichtliche Darstellung nicht mehr zu denken.

Kapitel 7

Das Verhalten einzelner Objekte

Während im letzten Abschnitt bei der interaktionsorientierten Spezifikation (siehe Kapitel 6 auf Seite 77) zwar die Objekte zu den Nutzungsfällen ermittelt wurden und eine Spezifikation des dynamischen Verhaltens in Zusammenhang mit anderen im Nutzungfall involvierten Objekten stattfand, wurde eine detaillierte Dynamikbeschreibung zu den einzelnen Objekten noch nicht geliefert. Diese Verhaltensbeschreibung der Objekte steht in diesem Kapitel im Vordergrund.

Als Beschreibungsmittel für das dynamische Verhalten werden sogenannte Statecharts eingesetzt, die ein Objekt als endlichen Automaten mit Zuständen und Zustandsänderungen beschreiben. Die SCs erlauben eine Verhaltensbeschreibung der betrachteten Objekte nicht nur auf exemplarischer Ebene, sondern erlauben eine vollständige Beschreibung der dynamischen Zustandsänderungen auf unterschiedlichen Detaillierungsebenen.

Eine Einführung zu Statecharts im Zusammenhang mit der Verhaltensbeschreibung wird im Abschnitt 7.1 gegeben. Im Abschnitt 7.2 werden im Sequenzdiagramm zum Nutzungsfall Büchersuche am Vorgangsobjekt die Objektzustände ermittelt und daraus das Statechart durch Einführung der Transitionen entwickelt. Das entstandene Statechart wird durch wiederholte Verfeinerungen mit zusätzlichen Detail–Informationen angereichert. Das Sitzungsobjekt, welches den Ablauf der Kunden–Sitzung steuert, stellt den Mittelpunkt vom Abschnitt 7.3 dar. Bestellungen wird bezüglich dem Grad ihrer Abarbeitung ein Status zugeteilt. Die Veränderung des zugehörigen Statusattributs mit seinen möglichen Werten wird im letzten SC dieses Kapitels dargestellt, siehe hierzu Abschnitt 7.4. Eine Zusammenfassung findet im Abschnitt 7.5 statt.

7.1 Einführung zu Statecharts

Statecharts beschreiben in dieser Arbeit einzelne Objekte, als Automaten mit Zuständen und Zustandsänderungen. Die letzteren werden durch das Eintreffen von Nachrichten angestoßen.

Ein Statechart ordnet einem Objekt eine endliche, und im allgemeinen kleine Menge von Zuständen zu, sogenannte Automatenzustände. Die Zustände des Automaten bieten eine abstrakte Sicht auf die konkreten Objektzustände und fassen meist viele Objektzustände zusammen, die bezüglich des Empfangs und Sendens von Nachrichten ähnliches Verhalten zeigen.

Objekte ändern ihre Zustände durch das Empfangen und Verarbeiten von Nachrichten. Zur Präsentation von Zustandsänderungen werden in den Diagrammen Pfeile verwendet. Sie sind mit Nachrichten attributiert, die ein Objekt empfangen kann. Als Reaktion auf eingehende Nachrichten können Objekte Nachrichten an andere Objekte verschicken. Die Zustandsänderungen werden auch als Transitionen bezeichnet. Neben Zuständen und Transitionen enthalten die Statecharts spezielle Symbole für Start und Ende.

Statecharts beschreiben Folgen von Nachrichten, die das spezifizierte Objekt empfangen kann, sowie die, Reaktion auf die empfangenen Nachrichten, nämlich die ausgesendeten Nachrichten und die Zustandsänderungen des Objekts. Im Unterschied zu (einfachen) Sequenzdiagrammen sind Statecharts eine Beschreibungstechnik, mit der ein Objektverhalten vollständig spezifiziert werden kann.

Nicht für jede Nachricht muß in jedem Zustand eine geeignete Transition zur Verfügung stehen. In diesem Fall ist das Statechart unterspezifiziert und kann durch spätere Verfeinerungen geeignet um Transitionen gerweitert werden.

Zusätzlich können existierende Transitionen in ihrem Effekt unterschiedlich detailliert beschrieben sein, oder mehrere Transitionen zur Auswahl stehen. Auch diese Form der Unterspezifikation kann durch Verfeinerungstransformationen verkleinert werden und so zu detaillierteren Beschreibungen führen.

Die Statecharts können durch die Anreicherung um Zustände aus den Sequenzdiagrammen gewonnen werden. Das exemplarische Objektverhalten muß dann oftmals noch mit weiteren Zuständen und Transitionen ergänzt werden, die z.B. aus einem anderen Nutzungsfall stammen. Da in dieser Fallstudie nicht alle Nutzungsfälle ausgearbeitet wurden, müssen genau wie in der Praxis auch eventuell fehlende Transitionen intuitiv ergänzt werden. In [KGSB99] wird eine Vorgehensweise zur Transformation von einem MSC in ein STD beschrieben. Daneben können auch Statecharts ausgehend von Statusattributen mit den zulässigen Werten in Objekten erstellt werden. Da solche Statusattribute mit einer Menge von möglichen Werten belegt werden können, können die Belegungen als Zustände in einem Automat dargestellt werden. Die Nachrichten, die eine Veränderung der Attributbelegung hervorrufen, werden an den Transitionen vermerkt. Eine solches Diagramm beschreibt das dynamische Verhalten eines Objekts mit abstrakten Zuständen, die in diesem Fall den Werten von Statusattributen entsprechen.

Statecharts sind ebenso wie die bereits verwendeten Klassen–, Nutzungsfall– und Sequenzdiagramme Bestandteil der UML. Eine Beschreibung beziehungsweise Anwendung der

Statecharts ist in [Rum98b], [Rum98a], [PR97], [KPR97], [KRB96], [Rum96], [BBH+99], [BGH+98], [BHH+97], [Bre98a], [Bre98b], [BRJ98], [BRS97], [FS98], [JBR98], [KGSB99], [Oes97] und [Rat97] zu finden.

7.2 Das Vorgangsobjekt zur Büchersuche

Wie bereits im Teilabschnitt 6.5.4 auf Seite 113 erklärt wurde, handelt es sich bei der Fachklasse Suche um eine Superklasse für die „Einfache Suche“, die „Zusammengesetzte Suche“ und die „Expertensuche“. Diese Beziehung wird auch im Klassendiagramm zur Büchersuche, siehe Abbildung 8.6 auf Seite 142 dargestellt. In diesem Abschnitt wird zuerst das Vorgangsobjekt zur abstrakten Büchersuche, d.h. das Verhalten des Objekts als Instanz der Superklasse analysiert (siehe Teilabschnitt 7.2.1). Im Teilabschnitt 7.2.2 werden dann Teile davon für eine „Zusammengesetzte Suche“ verfeinert.

7.2.1 Verhaltensanalyse der Suche

Das exemplarische Sequenzdiagramm mit seiner zusätzlichen informellen Beschreibung, die insbesondere auf mögliche Alternativen eingeht (siehe hierzu Teilabschnitt 6.5.3 auf Seite 111), wird zur Verhaltensbeschreibung des Objekts „VgBüchersuche“ herangezogen. Das Vorgangsobjekt zur Büchersuche wurde dort bereits eingeführt und dessen dynamisches Verhalten wird nun an dieser Stelle mit Hilfe eines Statecharts vollständig beschrieben.

Das Statechart zeigt die Zustandsänderungen in Abhängigkeit von eingehenden Nachrichten. In solchen Diagrammen findet keine Darstellung dahingehend statt, von welchem kommunizierenden Objekt die transitionsauslösende Nachricht ausgesendet wird.

Zur Erstellung des Statecharts wird im bereits existierenden exemplarischen Sequenzdiagramm aus Abbildung 6.27 auf Seite 112 nach [KGSB99] nur noch das Vorgangsobjekt „VgBüchersuche“ mit seinen ein- und ausgehenden Nachrichten betrachtet. Zu Beginn werden in den exemplarischen Ablauf abstrakte Zustände eingefügt. Dazu wird der Vorgang in bezug auf ein Objektes analysiert. An Stellen, an denen objektspezifische Bedingungen erfüllt sind, werden Zustände dem Objekt zugeordnet und in die Lebenslinie eingezeichnet. Diese Zustände werden anschließend in ein Statechart aufgenommen. Neben diesen Zuständen erfolgt die Aufnahme eines Start– und Endesymbols sowie der sequentiellen Transitionen aus dem Sequenzdiagramm. Die Beschriftungen an den Transitionen spiegeln entweder den Bezeichner der transitionsauslösenden Nachricht oder einen geeigneten Bezeichner im Fall von mehreren Einzelnachrichten wider.

Im exemplarischen Sequenzdiagramm zur Büchersuche wurden dem Vorgangsobjekt acht abstrakte Zustände zugewiesen. Abbildung 7.1 auf der nächsten Seite zeigt das Diagramm.

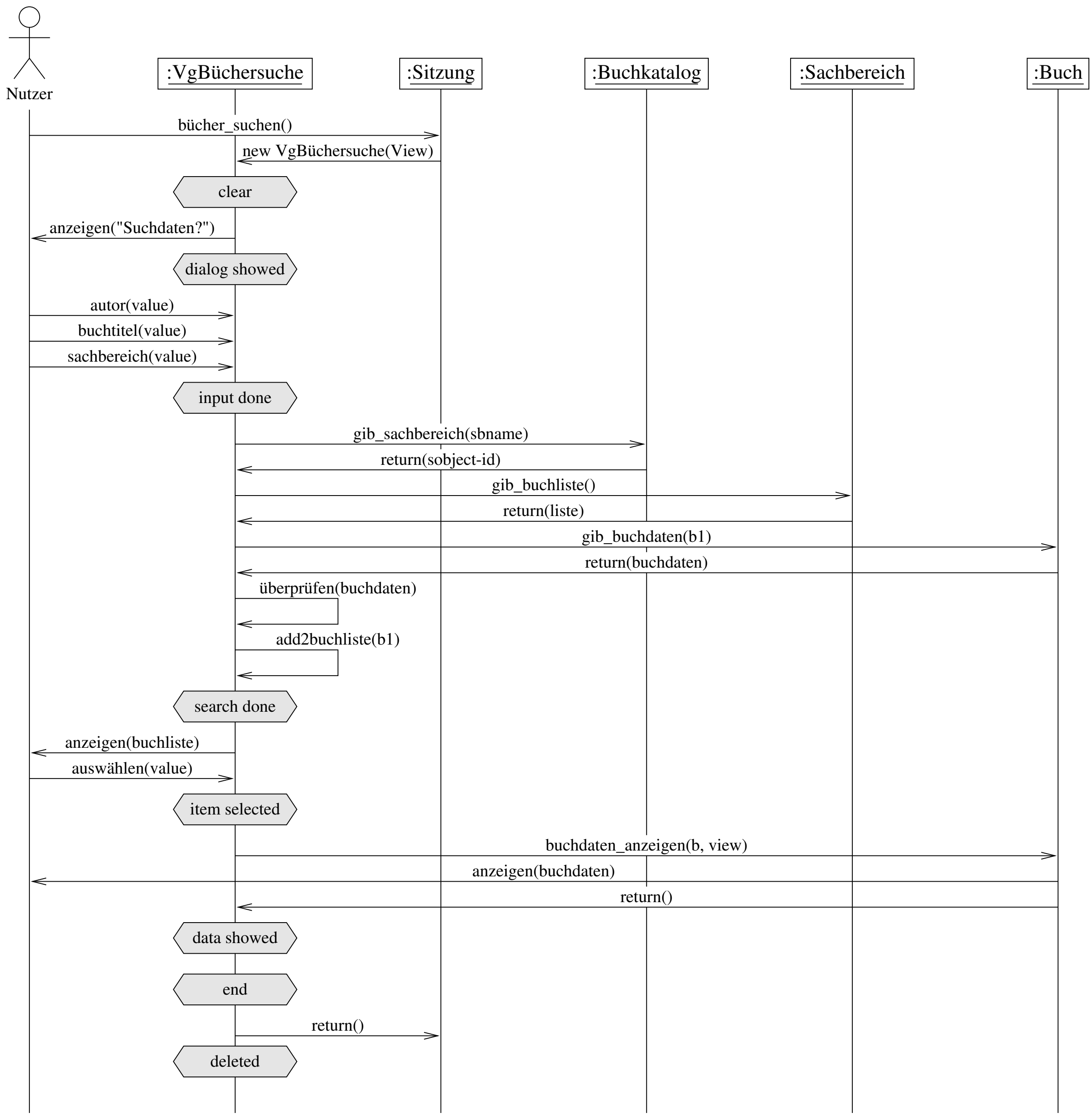


Abbildung 7.1: Sequenzdiagramm zur Büchersuche mit Zuständen

Die grau hinterlegten Sechsecke stellen die Zustände im Sequenzdiagramm dar. Die Syntax der ungleichseitigen Sechsecke wurde aus [KGSB99] übernommen, während die graue Hinterlegung nur zur Hervorhebung der veränderten Stellen im Diagramm dient.

Im Zustand „clear“ befindet sich das Vorgangsobjekt zur Büchersuche nach seiner Erzeugung. Kein Attribut ist besetzt. Nach der Aufforderung zur Suchdateneingabe nimmt das Objekt den Zustand „dialog showed“ und nach deren Eingabe „input done“ an. Der komplexe Suchvorgang wurde nicht weiter untergliedert, so daß nach der Suche „search done“ und nach der Auswahl eines Eintrags zur genaueren Betrachtung „item selected“ erreicht wird. Die Zustände „data showed“ und „end“ werden nach der Buchdatenanzeige eingenommen, bevor das Objekt in den abstrakten Zustand „deleted“ übergeht.

Diese neu erstellten Zustände werden direkt in das Statechart übernommen. Vor dem ersten Zustand und nach dem letzten wird das Start– bzw. Endesymbol plaziert und die Transitionen für den sequentiellen Ablauf können dazwischen übernommen werden. Die Beschriftungen spiegeln auf abstrakter Ebene die Nachrichtenflüsse zwischen den Zuständen im Sequenzdiagramm wider. Weiterhin werden in das Statechart noch Transitionen zum Neustart der Sucheingabe und zum vorzeitigen Beenden an geeigneten Stellen eingefügt. Diese Information war im ursprünglichen MSC nicht vorhanden. Die hinzugenommenen Transitionen beschreiben das Verhalten stärker detailliert und verfeinern somit das Statechart, das aus dem MSC destilliert worden ist. Da die Büchersuche eventuell eine Vielzahl von Büchern liefert, kann zu jedem Listeneintrag eine genaue Buchinformation angefordert werden. Dies impliziert im Statechart eine rückwärtsgerichtete Kante vom Zustand „data showed“ zu „search done“. Abbildung 7.2 zeigt das entstandene Statechart.

Das entstandene Statechart weist einige einheitliche Transitionen „Ende“ und „neue Abfrage“ auf. Mehrere Zustände werden deshalb in Abbildung 7.3 in einen übergeordneten Zustand „dialog“ gruppiert und Transitionen entsprechend angepasst.

In diesem Statechart kann nun das gesamte Objektverhalten in dieser Abstraktionsstufe nachvollzogen werden. Zusätzlich zu den vorhandenen abstrakten Zuständen können auch weitere hinzugefügt werden, so daß die Beschreibung des Objektverhaltens noch spezieller wird. Werden in ein Diagramm jedoch sehr viele Zustände eingefügt, so kann dies zu sehr vielen Transitionen und damit zu unübersichtlichen Diagrammen führen. Zu Transitionen können in Statecharts auch Vor– und Nachbedingungen sowie ausgehende Nachrichten hinzugefügt werden, was jedoch in dem gerade erstellten Diagramm nicht notwendig war.

7.2.2 Verhaltensanalyse der zusammengesetzen Suche

Ausgehend vom im vorausgehenden Teilabschnitt erstellten Statechart zur Büchersuche und dem exemplarischen Sequenzdiagramm aus dem Teilabschnitt 6.5.3 auf Seite 111 mit der zugehörigen verbalen Beschreibung werden in diesem Teilabschnitt eine Automatenverfeinerung entlang der Klassenhierarchie durchgeführt. Dabei werden die zwei Transitionen

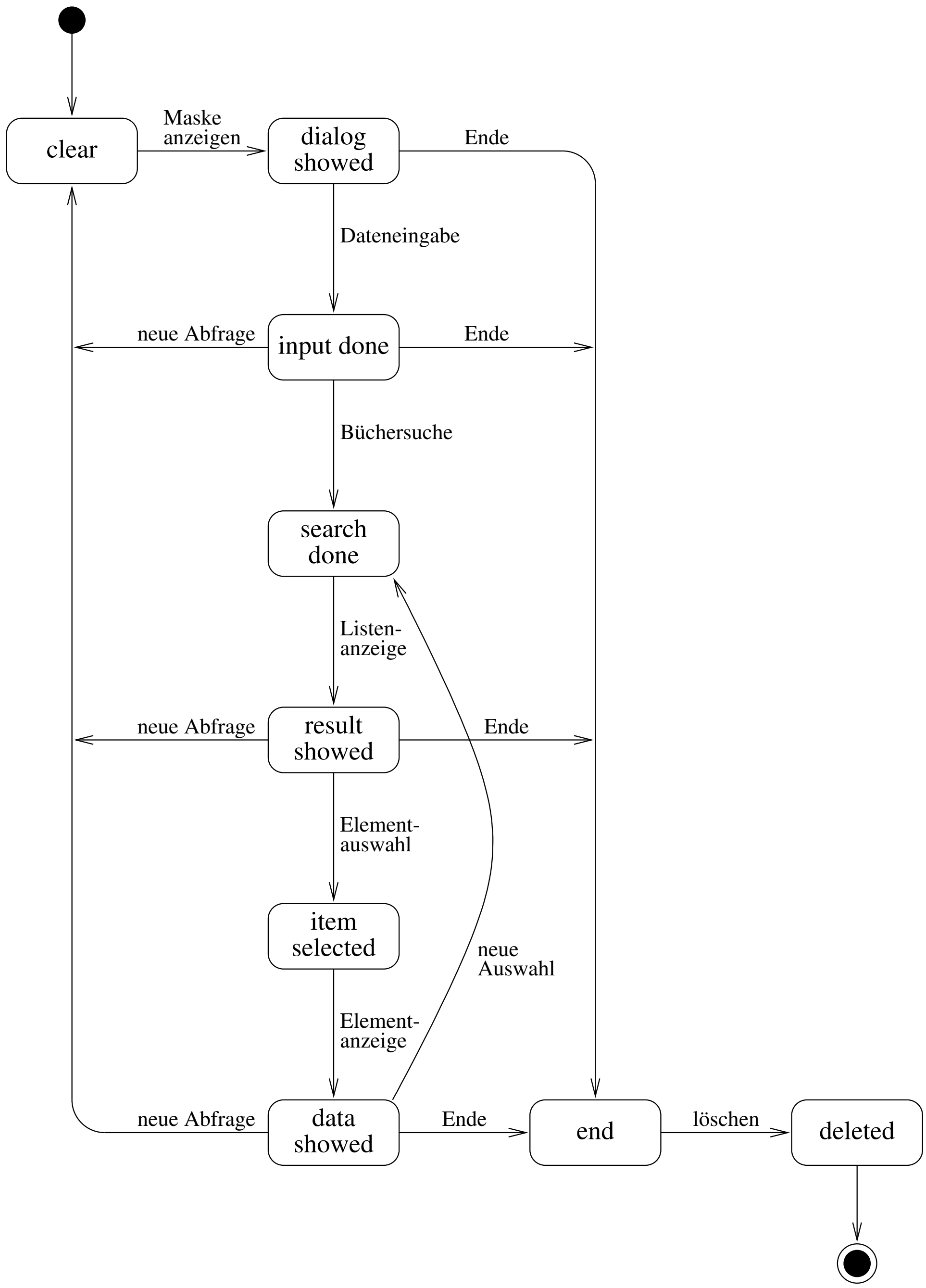


Abbildung 7.2: Statechart zur Büchersuche

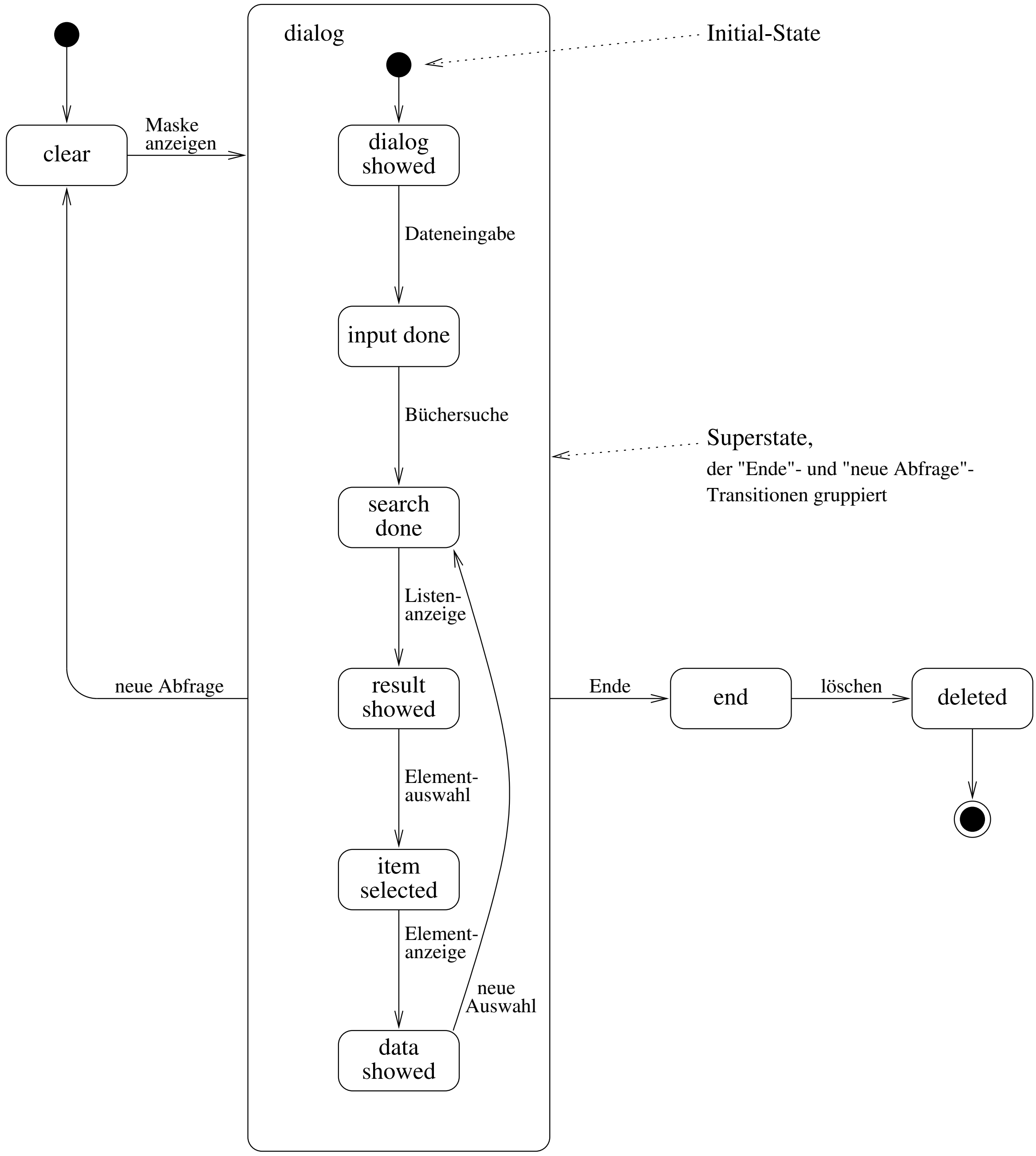


Abbildung 7.3: Statechart zur Büchersuche mit übergeordnetem Zustand

„Maske anzeigen“ und „Dateneingabe“ für den speziellen Fall der zusammengesetzten Suche um weitere Zustände und Transitionen angereichert. Das verfeinerte Statechart wird von der Klasse „Suche“ auf die Subklasse der Klasse „Zusammengesetzte Suche“ vererbt (vgl. Abbildung 8.6 auf Seite 142). Bei der zusammengesetzten Suche werden die Suchdaten z.B. für den Autor und den Titel in zwei getrennte Textfelder eingegeben, während diese bei der einfachen Suche gemischt in ein einziges Textfeld eingegeben werden. In der folgenden Betrachtung besteht die zugehörige Eingabemaske aus drei Textfeldern. Der Autor, der Titel und das Erscheinungsjahr können angegeben werden. Für eine vollständige Eingabe wären weitere Felder möglich, z.B. der Buchverlag. Da dies jedoch nur die Zustandsanzahl erhöhen und die Diagramme in die Länge ziehen würde, aber andererseits keine weiteren Neuheiten liefern würde, werden hier exemplarisch nur die drei erwähnten Eingabefelder betrachtet.

An zwei Diagrammen wird im folgenden die Einführung von neuen Zuständen mit Transitionen gezeigt. Außerdem werden für die Transitionen Vorbedingungen eingeführt. Ausgehend von einem Zustand kann dann eine Transition nur durchgeführt werden, wenn die Vorbedingung für diese Transition erfüllt ist. Die Angabe der Vorbedingung erfolgt neben dem Transitionsbezeichner. Sie wird dabei in eckige Klammern gesetzt (vgl. [BBH+99]).

Abbildung 7.4 zeigt die Verfeinerung der Transition „clear“ → „dialog showed“ aus dem Statechart zur Büchersuche (siehe Abbildung 7.2), in dem sechs neue Zustände und die notwendigen Transitionen hinzugefügt wurden. Die bereits bestehenden Zustände wurden für eine übersichtlichere Darstellung grau hinterlegt. Nachdem das Objekt die Nachricht zum Erzeugen der Maske enthalten hat, erfolgt zuerst die Fenstererstellung, die Ausgabe von Textfeldern für die Dateneingabe und anschließend die Ausgabe des Beschreibungstextes in der Maske. Nach jeder Darstellung wird bezüglich eines Abbruchs geprüft, z.B. von der Seite des Anwenders. Abbildung 7.4 zeigt nur einen Teilausschnitt des Statecharts zur zusammengesetzten Suche. Die grau hinterlegten Zustände sind bereits bekannte aus dem Statechart zur Büchersuche (vgl. Abbildung 7.2 auf Seite 124).

In Abbildung 7.5 findet die Erweiterung der Transition „dialog showed“ statt. Die grau hinterlegten Zustände stellen hier ebenfalls die bereits bekannten aus dem Statechart zur Büchersuche dar, während sieben neue Zustände mit ihren Transitionen hinzugefügt wurden. In diesem Diagramm kommen Vorbedingungen zum Vorschein. Falls z.B. bei der Sucheingabe zuerst der Titel eingegeben wird, erfolgt die Transition „Eingabe“ mit der Bedingung „Titel“. Nachfolgend können weitere Eingaben erfolgen oder die Eingabe abgeschlossen werden. Es sind nun alle Reihenfolgekombinationen für die Eingabe von Suchdaten erlaubt. So kann z.B. zuerst die Maske für das Erscheinungsjahr, dann der Titel und zuletzt der Autor angegeben werden. Die verschiedenen Möglichkeiten können im Diagramm nachvollzogen werden.

Bezüglich des Abbruchverhaltens wurden im erweiterten Eingabediagramm keine Zustandsübergänge eingeführt. Diesbezüglich ist das erstellte Statechart unvollständig. Die dadurch entstandene Unterspezifikation ist notwendig, weil hier zunächst kein allgemeines Verhalten

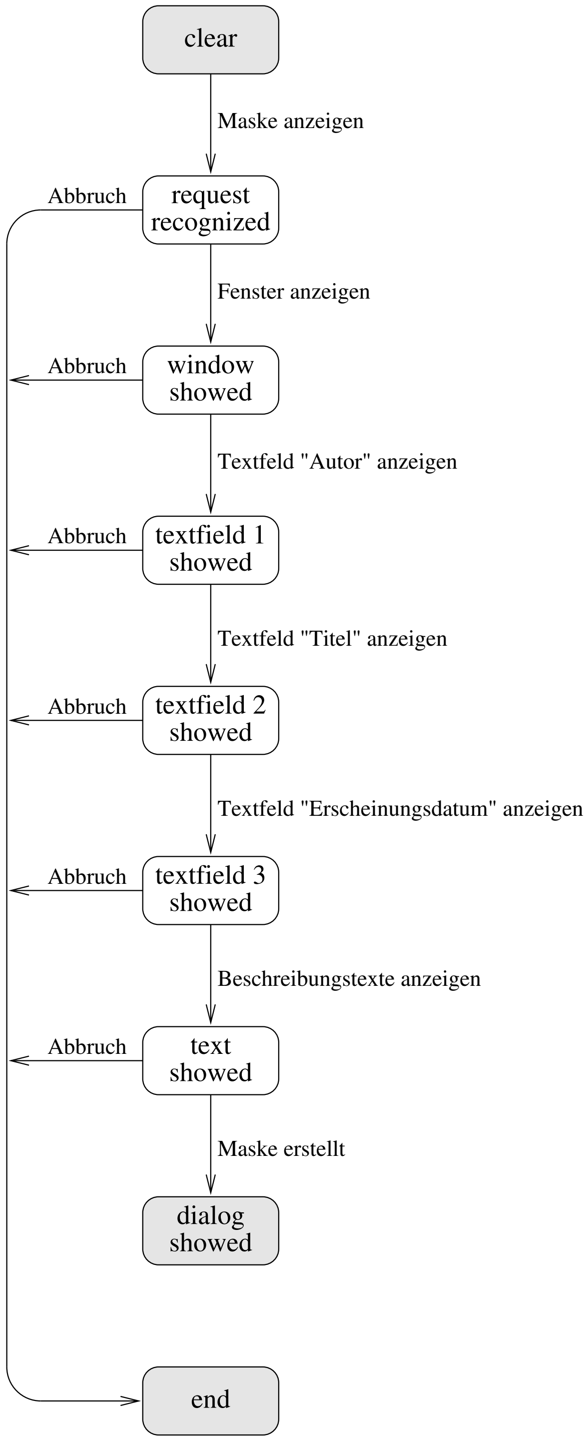


Abbildung 7.4: Erweiterte Dialoganzeige im Statechart zur zusammengesetzten Suche

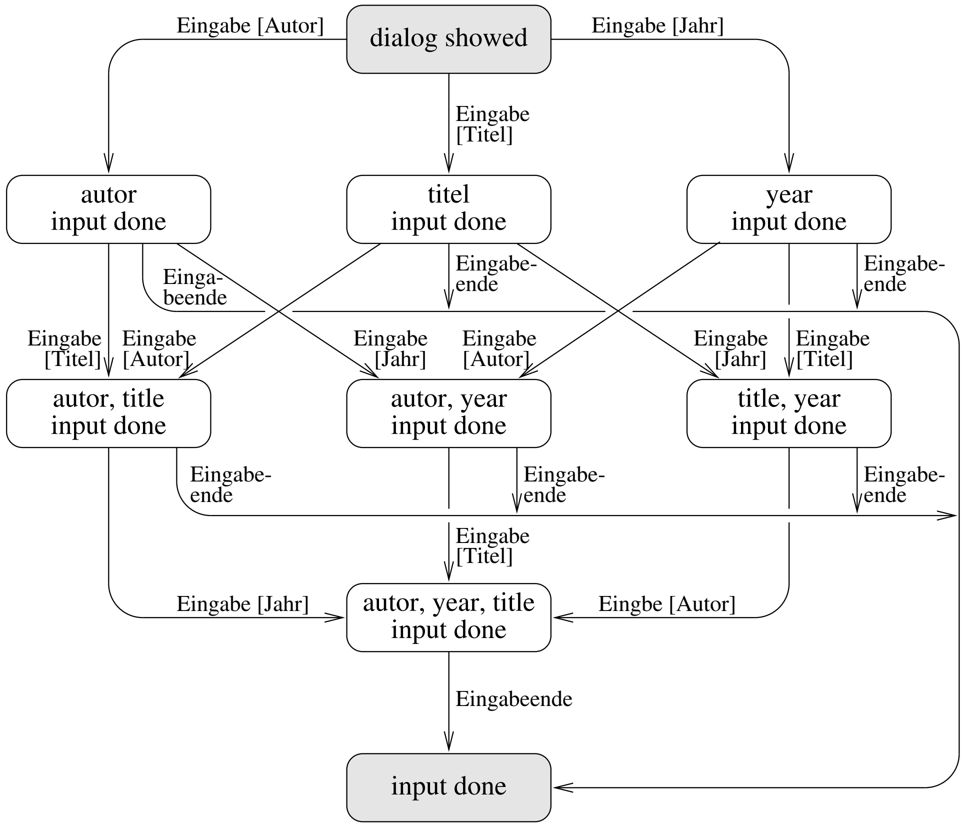


Abbildung 7.5: Erweiterte Eingabe im Statechart zur zusammengesetzten Suche

vorausgesetzt wird, da prinzipiell alles möglich ist (Chaos). Für eine detaillierte Spezifikation ist eine Verfeinerung des Automaten notwendig.

7.3 Das Sitzungsobjekt

Das Sitzungsobjekt steuert den gesamten Ablauf einer Kundensitzung. Die zugehörigen Fachklassen wurden bei der Entwicklung des Anwendungskerns im Teilabschnitt 4.1.2 auf Seite 23 bereits ermittelt. Im Kapitel 6 erzeugte das Sitzungsobjekt für die verschiedenen Nutzungsfälle der Händlersitzung das jeweils notwendige Vorgangsobjekt. Da in einer Sitzung der Kunde Tätigkeiten sowohl als unautorisierter Benutzer als auch als bekannter Anwender vollbringen kann, wird ein Statechart bezüglich der Nutzeridentifikation und den ausgeführten Tätigkeiten entwickelt.

In dieser Betrachtung erfolgt nur eine Unterscheidung zwischen zwei Zuständen zur Identifikation des Kunden. Bezüglich des Eingabeverhaltens oder eines anderen Ablaufverhaltens soll der Automat keine Auskunft geben. Aus diesem Grund werden für das folgende

Statechart keine Zustandsverfeinerungen durchgeführt. Da auch nur bezüglich der durchgeführten Tätigkeiten auf der Ebene von Methoden im Automaten unterschieden wird, werden keine Vor– und Nachbedingungen benötigt.

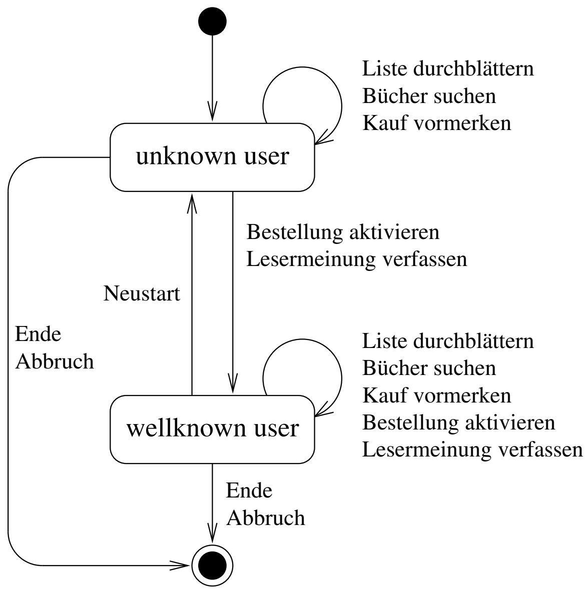


Abbildung 7.6: Das Statechart zum Sitzungsobjekt

In das Statechart werden nur die beiden Zustände „unknown user“ und „wellknown user“ aufgenommen. Solange ein Kunde im Bookshop–Katalog nur nach Büchern sucht oder diesen durchblättert sowie irgendwelche Buchartikel im virtuellen Einkaufskorb ablegt, muß vom System seine Identität nicht festgestellt werden, da diese Dienste kostenlos sind. Der Kunde kann das Programm jederzeit abbrechen, auch wenn er schon einige Artikel im Einkaufskorb abgelegt hat. Da diese keinem Kunden zugeordnet werden können, gehen sie verloren. Für diese Abläufe sind zwei Transitionen notwendig. Solange der Anwender Artikel sucht, vormerkt oder im Katalog blättert, bleibt er im Zustand „unknown user“. Für einen Abbruch führt eine Transition aus diesem Zustand zum Ende–Symbol.

Will der Kunde nun eine Bestellung aktivieren oder eine Lesermeinung zu einem Artikel verfassen, so wird hier seine Identität festgestellt, worauf er in den Zustand „wellknown user“ wechselt. In diesem Zustand kann er beliebig oft Such–, Bestell– und Kommentar–Aktionen durchführen. Beantragt der Kunde einen Programmneustart, wechselt er wieder zum „unknown user“. Das normale Programmende ist ebenso möglich. Abbildung 7.6 zeigt das zugehörige Statechart.

Dieses relativ einfache Transitionsdiagamm zeigt das Verhalten des Sitzungsobjekts mit den abstrakten Zuständen bezüglich der Nutzeridentität. Hieraus kann abgelesen werden, welche Aktionen bezüglich des Bekanntheitsgrades möglich sind und wann zwischen den Bekanntheitszuständen gewechselt wird.

7.4 Die Veränderung der Statuswerte zur Bestellung

Aus der Einführung des ordinalen Status–Typs zur Bestellung (siehe hierzu Seite 39) ergibt sich, daß sich die Veränderung der Bestellwerte in einer Bestellung in einem Automaten darstellen läßt. Die Zustände des Typs und die nötigen Transitionen sind in Abbildung 7.7 dargestellt.

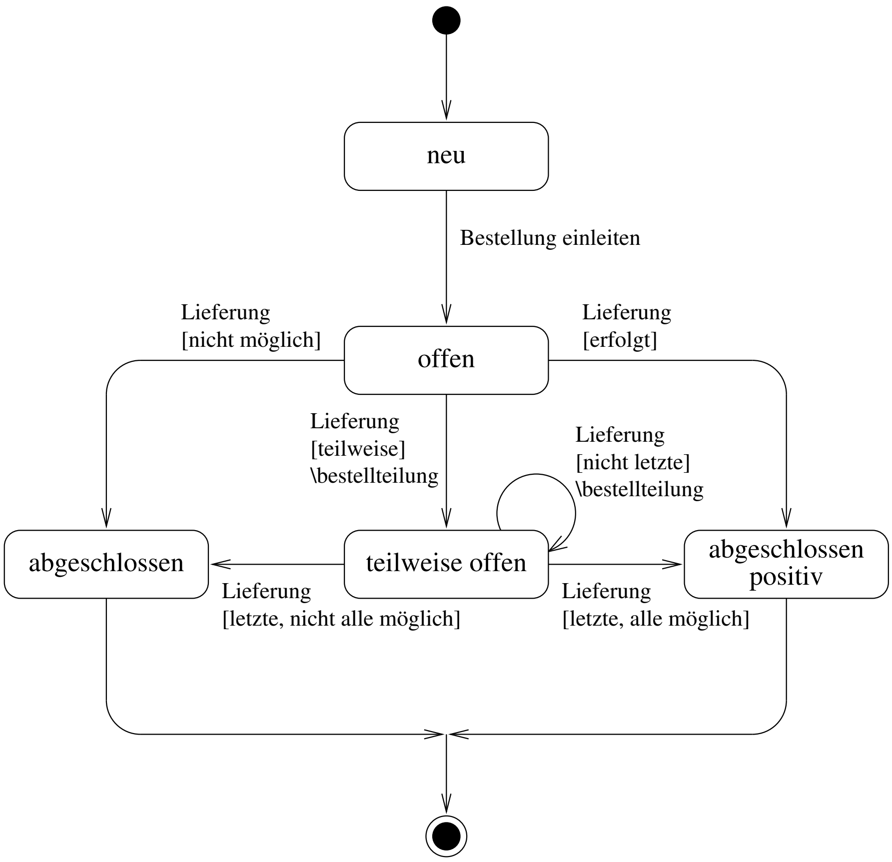


Abbildung 7.7: Das Statechart zum Übergang der Statuswerte

Im Gegensatz zu den bisher besprochenen Automaten beschäftigt sich dieser nicht mit dem Verhalten eines gesamten Objekts, sondern nur mit der Zustandsänderung des Statusattributs der Bestellung. Er beschreibt damit eine auf das Statusattribut bezogene Abstraktion des Objektzustandsraums. Da alle möglichen Statuswerte bereits festgelegt und auch die Transitionen bereits vollständig erfaßt wurden, wird dieses Statechart nicht aus Sequenzdiagrammen, sondern direkt aus dem beschriebenen Anforderungskatalog destilliert. abweichend zu den bisherigen behandelt. Ausgehend vom dargestellten Statechart werden die in den Kapitel 5 und 6 erstellten Szenarien analysiert, um mögliche Verbindungen zu den Bestellübergängen zu erforschen.

Die Ermittlung von Beziehungen auf diese Art und Weise liefert eine Beziehung zwischen

Transitionen und Nutzungsfällen, denn das dynamische Verhalten in den Nutzungsfällen spezifiziert auch in den relevanten Stellen die Bestellstatus–Übergänge.

Nachdem der Kunde die ausgewählten Bücher im Einkaufskorb abgelegt hat, werden diese bei der definitiven Bestellung einem neu erstellten Bestellungsobjekt hinzugefügt. Bei der Entwicklung des Szenarios „Bestellung der im Einkaufskorb liegenden Bücher“ auf Nutzungsfallebene, siehe hierzu Teilabschnitt 6.4 auf Seite 89, wurde die Nachricht zum Erzeugen einer Bestellung im Sequenzdiagramm dargestellt (vgl. Abbildungen 6.11 auf Seite 95 und 6.15 auf Seite 100). Bei der Erstellung wird das Statusattribut implizit mit „neu“ belegt. Abbildung 7.8 zeigt den Ausschnitt aus dem Sequenzdiagramm, bei dem der abstrakte Zustand in der Objekt–Lebenslinie ergänzt wurde. Zur Hervorhebung der Ergänzung ist der neue Zustand grau hinterlegt.

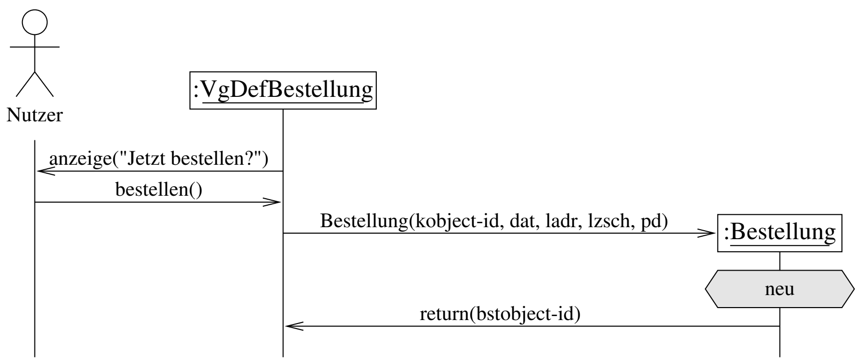


Abbildung 7.8: Sequenzdiagramm zur Erstellung eines Bestellobjekts

Eine Bestellung kann auch vom Händler–Personal manuell angelegt werden, wie dies bereits im Systemszenario „Bestellung hinzufügen“ in Abbildung 5.7 auf Seite 65 dargestellt wurde. Abbildung 7.9 auf der nächsten Seite zeigt den relevanten Ausschnitt dieses Systemszenarios. Dabei wurde wieder der Zustand des Bestellobjekts ergänzt. Da sich der Zustand aber speziell auf das Bestellungsobjekt bezieht und hier nur das abstrakte Systemobjekt abgebildet ist, wurde im Zustandssymbol die Abkürzung BS für den Bestellzustand eingefügt.

Wird eine neu eingetragene Bestellung in den Abarbeitungsprozeß übernommen, so wird der Statusattribut mit dem Wert „offen“ belegt. Falls das Buch im Lager vorrätig ist, wird es sofort ausgeliefert und ein Statuswechsel zu „abgeschlossen positiv“ im Fall einer Auslieferung bzw. zu “abgeschlossen“ bei einer Nicht–Auslieferung findet statt. In Teilabbildung 5.8 l: „Lageraufträge erstellen“ auf Seite 66 ist dieser Ablauf implizit enthalten, falls die Artikel im Lager vorrätig sind. Ein Hinweis auf eine Bestellstatusänderung wurde dort nicht explizit gegeben, kann aber aus dem Sequenzdiagramm zum Systemszenario bei genauer Analyse jederzeit abgeleitet werden.

Müssen die Artikel von einem bzw. mehreren Vertriebspartnern angefordert werden, nimmt das Statusattribut den Wert „offen“ an und wechselt diesen erst, wenn der Wareneingang

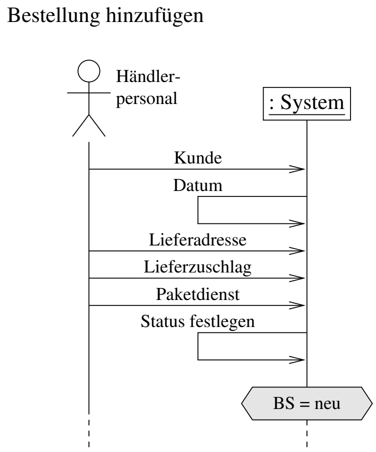


Abbildung 7.9: Sequenzdiagramm zum manuellen Hinzufügen einer Bestellung (Ausschnitt)

übernommen wird und die Lageraufträge hierzu erstellt werden. Mögliche Werte sind dann wieder „abgeschlossen“ und „abgeschlossen positiv“. Abgeschlossen bedeutet hier, daß sich im Verlauf der Vertriebsbestellung gezeigt hat, daß ein bestellter Artikel nicht geliefert werden kann. In den Systemszenarien „Vertriebsbestellung aktivieren“ und „Wareneingang übernehmen“ aus der Bestellverwaltung (siehe Abbildung 5.8 auf Seite 66) wurden bereits die abstrakten Nachrichten „Bestellstatus ändern“ eingefügt. Abbildung 7.10 zeigt diese Sequenzdiagramme mit den analog zur Abbildung 7.9 eingefügten Bestellzuständen.

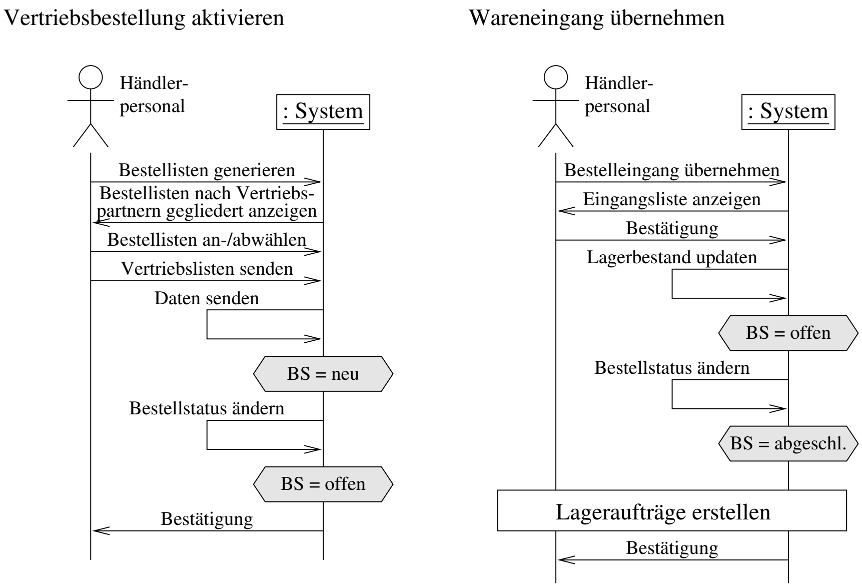


Abbildung 7.10: Sequenzdiagramm zum Aktivieren der Betriebsbestellung und Übernehmen des Wareneingangs

Kann nur ein Bestellungsteil abgearbeitet werden, so teilt sich die Bestellung (siehe Seite 39). Bei der aktuellen Bestellung wird der Statuswert „teilweise offen“ eingetragen und sie enthält, wie beschrieben, zwei untergeordnete Teilbestellungen. Bei diesen Teilbestellungen wird entsprechend ihrer Abarbeitung wieder der Bestellstatus gesetzt bzw. modifiziert. Der Bestellstatus einer übergeordneten Bestellung behält dann so lange den Wert „teilweise offen“, bis die letzte Teilbestellung abgeschlossen wird. Sobald entweder der Status „abgeschlossen“ bzw. „abgeschlossen positiv“ erreicht wird, verändert sich am Statusattribut nichts mehr. Dies ist im Statechart mit einer Transition zum Symbol für das Bearbeitungsende dargestellt.

Ausgehend vom intuitiv entwickelten Statechart wurden die Transitionen „neu“ → „offen“, „offen“ → „abgeschlossen“ und „offen“ → „abgeschlossen positiv“ neben der Belegung des Statusattributs bei der Erzeugung des Bestellobjekts ermittelt. Die Transitionen vom und zum Zustand „teilweise offen“ waren aus den Szenarien auf Systemebene und auf Nutzungsfallebene nicht ersichtlich. Dies ergibt sich daraus, daß in den verwendeten Szenarien immer Zustände von Objekten am Ende eines Pfades betrachtet wurden, denen keine Teilbestellungen mehr zugewiesen sind. Es handelt sich hierbei um sogenannte Blattknoten im Baum zur rekursiven Teilbestellung. Die notwendigen Transitionen zur teilweisen Abarbeitung gehen jedoch aus dem Zusammenhang und vor allem aus der Einführung zur rekursiven Teilbestellung hervor (siehe Abschnitt 4.2).

7.5 Zusammenfassung

Das Verhalten einzelner Objekte wurde ausgehend von exemplarischen Sequenzdiagrammen inklusive verbaler Anmerkungsbeschreibung, von Methodenaufrufen in Nutzungsfällen und von den möglichen Zuständen eines aufzählbaren Typs erstellt.

Im ersten Teilabschnitt dieses Kapitels wurde zum Suchobjekt ein Vorgangsobjekt ausgehend von exemplarischen Sequenzdiagrammen erstellt. Hierzu fand eine vorausgehende Anreicherung des Sequenzdiagramms mit Objektzuständen statt. Transitionen wurden entsprechend dem beispielhaften Ablauf und bezüglich Ablaufvarianten neben den festgestellten Zuständen im Statechart eingefügt und das Statechart so iterativ verfeinert. Mehrere Einzelnachrichten wurden dabei zu einer Transition zusammengefaßt.

Da die Suche nur das Grundverhalten von der einfachen, der erweiterten und der zusammengesetzten Suche darstellt, wurde das Verhalten der speziellen Suchoperationen in ihren eigenen Objekten noch verfeinert und ergänzt. Dabei handelt es sich um eine Automatenverfeinerung, die zur Vererbung von Verhalten entlang der Klassenhierarchie genutzt wird. Aus diesem Grund wurden zwei Transitionen für die zusammengesetzte Suche näher analysiert. Hierbei wurden zusätzliche Zustände und notwendige Transitionen eingeführt. Außerdem kamen Vorbedingungen zum Einsatz. Sie erlauben eine Transitionsauswahl, in Abhängigkeit einer erfüllten Bedingung.

Im Statechart zur erweiterten Eingabe (siehe Abbildung 7.5 auf Seite 128) wurden keine Programmabbrüche im Verhalten spezifiziert. Statecharts dürfen in ihrer Darstellung unvollständig sein, d.h. es können gewisse Nachrichtenfolgen fehlen oder Zustandsänderungen nicht bis ins letzte Detail spezifiziert sein. Die Semantik des Automaten wird dadurch nicht vollständig spezifiziert (engl. „loose Semantik").

Der zweite Teilabschnitt behandelte ein Statechart, bei dem nur zwischen zwei internen Zuständen unterschieden wurde. Die möglichen Benutzeraktionen in Abhängigkeit von der Nutzeridentifizierung können in diesem Statechart abgelesen werden. Verfeinerungen und Vor– bzw. Nachbedingungen sind an diesem Statechart zur weiteren Analyse nicht notwendig.

Die möglichen Belegungen einer Statusvariablen bildeten im dritten und letzten Statechart den Ausgangspunkt für die Festlegung der Zustände und Transitionen. Ausgehend von der Definition der Statuswerte wurde dabei das Statechart erstellt und daraufhin mögliche Beziehungen in den Sequenzdiagrammen sowohl auf Systemebene sowie auf Nutzungsfallebene ermittelt. Da jedoch im Rahmen dieser Arbeit nicht alle Nutzungsfälle ausgearbeitet wurden, blieben zumindest einige Transitionen teilweise ohne vollständigen Zusammenhang zu den Szenarien. Bei einer vollständigen Bearbeitung hätten jedoch alle Transitionen aus den Nutzungsfällen ermittelt werden können.

Wie aus den verwendeten Beispielen hervorgeht, läßt sich das Verhalten von einzelnen Objekten sehr gut mit Automaten beschreiben. Der Anwender dieser Technik kann je nach gewünschtem Detaillierungsgrad entweder verschieden viele Objektzustände ermitteln oder die Statechart teilweise unspezifiziert lassen. Im zweiten Fall ist es dann jedoch möglich, daß das Verhalten nicht determiniert ist, während im ersten Fall nur die nötige Teilinformation aus dem Diagramm abgelesen werden kann. In den vorgestellten Anwendungen lief die Transformation zu den Automaten ohne Schwierigkeiten ab. Sollten die Diagramme jedoch zu groß werden, empfiehlt es sich, diese in Teildiagramme aufzugliedern, damit die Übersicht erhalten bleibt.

Kapitel 8

Die Klassenstruktur

Nachdem in den letzten drei Kapiteln (5 bis 7) die Beschreibung des Systemverhaltens im Vordergrund stand, findet zum Abschluß nochmals eine Betrachtung der statischen Klassenstruktur an ausgewählten Stellen statt.

An der Klasse Sitzung, die bereits im Fachklassendiagramm zum Einkaufskorb eingeführt wurde (siehe Teilabschnitt 4.1.2 auf Seite 23), findet im Abschnitt 8.2 eine Ergänzung von Attributen und Methoden statt. Im Abschnitt 8.3 wird die Vorgangsklasse zum definitiven Bestellen im Fachklassendiagramm aus Kapitel 4 auf Seite 17 um ihre zugehörigen Assoziationen ergänzt. In einem zweiten Schritt werden hier Aggregationen und Kompositionen in diesem neu erstellten Klassendiagramm analysiert. Außerdem werden hierbei zu einer Funktion die Vor– und Nachbedingungen spezifiziert. Für Vor– und Nachbedingungen, Aggregationen und Kompositionen findet im Abschnitt 8.1 eine Einführung statt. In den Abschnitten 8.4 und 8.5 wird jeweils ein Ausschnitt des Fachklassendiagramms mit den Vorgangsklassen zum Vormerken bzw. zur Büchersuche ergänzt. Abschnitt 8.6 liefert eine Zusammenfassung zu diesem Kapitel.

8.1 Aggregationen, Kompositionen, Vor– und Nachbedingungen

Nach [Bre98a] ist eine Aggregation eine IST–TEIL–VON–Beziehung. Sie besitzt keine Entsprechung in den Programmiersprachen und ist ein Konstrukt der Modellierung. Während in [Bre98a] nicht zwischen Aggregation und Komposition unterschieden wird, treffen wir hier die in [FS98] angegebene Unterscheidung. Dabei werden die IST–TEIL–VON–Beziehungen weiter unterteilt. Handelt es sich bei einer Instanz zu einer Klasse um ein Objekt, das nach dem Löschen des assoziierenden Objekts als eigenständiges Objekt weiter bestehen kann, so handelt es sich um eine Aggregation. Endet die Lebenzeit von beiden Objekten gleichzeitig, so nennen wir diese Beziehung Komposition. Weiterhin gilt für die Komposition, daß ein Teilobjekt höchstens einem Kompositionsobjekt zugeordnet sein kann. Eine Kapselung von Teilobjekten, wie sie in [Bre98a] angesprochen wird, kann sowohl bei der Aggregation als auch bei der Komposition vorliegen.

Durch Vor- und Nachbedingungen werden Objektzustände charakterisiert. Als Zustand eines Objekts wird hier die Attributbelegung und der Kontrollzustand, z.B. repräsentiert durch entsprechende Registerbelegungen, betrachtet (siehe hierzu das Systemmodell in [BBH+99]). Ist vor Ausführung einer Methode die spezifizierte Vorbedingung erfüllt, so wird die Gültigkeit der Nachbedingung nach Ausführung der Methode zugesichert.

Vor– und Nachbedingungen werden hier ebenfalls wie die Invarianten aus Kapitel 4 in der Sprache OCL definiert (siehe Abschnitt 4.2 auf Seite 34), die Bestandteil der UML ist (siehe hierzu [BBH+99] und [Rat97]).

8.2 Die Sitzung

In diesem Abschnitt werden Ergänzungen an der Klasse Sitzung erläutert, die im Fachklassendiagramm bereits eingeführt wurde. Diese haben sich im Verlauf der Betrachtung des Systems unter dynamischen Gesichtspunkten ergeben.

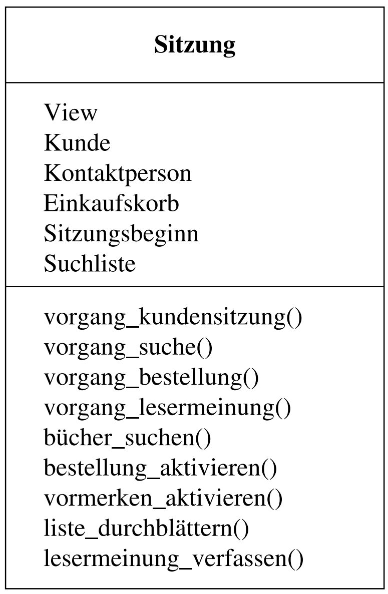


Abbildung 8.1: Klassendiagramm zur Sitzung

Im Fachklassendiagramm zum Einkaufskorb wurde die Klasse Sitzung eingeführt und dabei nur die beiden Assoziationen zum Kunden und zum Einkaufskorb festgelegt. Neben diesen

Attributen müssen auch noch Attribute für eine Kontaktperson, für das Ausgabeterminal, für den Sitzungsbeginn und für das Suchergebnis bereitgestellt werden.

Aus der Optimierung des Fachklassendiagramms zum Kunden (siehe Teilabschnitt 4.1.1 auf Seite 18) ergibt sich die Notwendigkeit, daß zu einem Kunden im Falle eines Firmenkunden eine Kontaktperson referenziert werden muß, da der Firmenkunde mehrere Kontaktpersonen beinhalten kann und zu einer Sitzung aber eine einzelne Kontaktperson referenziert wird.

In den Abschnitten 6.3.3 auf Seite 85, 6.4 auf Seite 89 und 6.5 auf Seite 109 wurden implizit die Methoden „vormerken aktivieren", „bestellung aktivieren" bzw. „bücher suchen" eingeführt, die hier nun ebenfalls im Klassendiagramm aufgeführt wurden. In Analogie zur Kunden–Sitzung aus Abbildung 5.2 auf Seite 57 müssen noch die vorgangsbeschreibenden Methoden „vorgang kundensitzung", „vorgang suche", „vorgang bestellung" und „vorgang lesermeinung" ergänzt werden, die sich direkt aus den Szenarien auf Systemebene ableiten. Die Methoden für die nicht ausgearbeiteten Nutzungsfälle zum Blättern im Katalog und zum Verfassen von Lesermeinungen, „liste durchblättern" und „lesermeinung verfassen" müssen auch noch mitaufgenommen werden. In Abbildung 8.1 auf der vorherigen Seite ist das Klassendiagramm zur Kundensitzung mit den ergänzten Attributen und Methoden zu sehen. Weiterhin können in der Sitzungsklasse noch Methoden enthalten sein, die eine Strukturierung der bereits genannten Methoden unterstützen. In diesem Entwurfsstadium berücksichtigen wir dies jedoch noch nicht.

8.3 Bestellung der im Einkaufskorb liegenden Bücher

Wie bereits in der Einleitung zu diesem Kapitel erklärt wurde, findet in diesem Abschnitt eine Ergänzung des Fachklassendiagramms mit der Vorgangsklasse zum Bestellen der im Einkaufskorb liegenden Bücher statt.

In dem in Kapitel 4 entwickelten Klassendiagramm aus den statischen Daten des Systems sind nur Fachklassen mit ihren Assoziationen enthalten. Bei der Ausarbeitung des Nutzungsfalls zum definitiven Bestellen, siehe Abschnitt 6.4 auf Seite 89, entstand die Vorgangsklasse „VgDefBestellung". Diese soll nun im Klassendiagramm ergänzt werden, was neben der Klassenergänzung eine weitere Einführung von Assoziationen und die Klärung von Vielfachheiten mit sich bringt.

Abbildung 8.2 auf der nächsten Seite zeigt den relevanten Ausschnitt aus dem Fachklassendiagramm inklusive der ergänzten Vorgangsklasse und der nötigen Assoziationen. Die Assoziationen wurden aus dem exemplarischen Sequenzdiagramm ermittelt. Über die Vielfachheiten konnten dem Sequenzdiagramm jedoch keine Angaben entnommen werden, so daß diese hier neu ermittelt werden mußten. Die ergänzte Vorgangsklasse wurde zur besseren Erkennung grau hinterlegt.

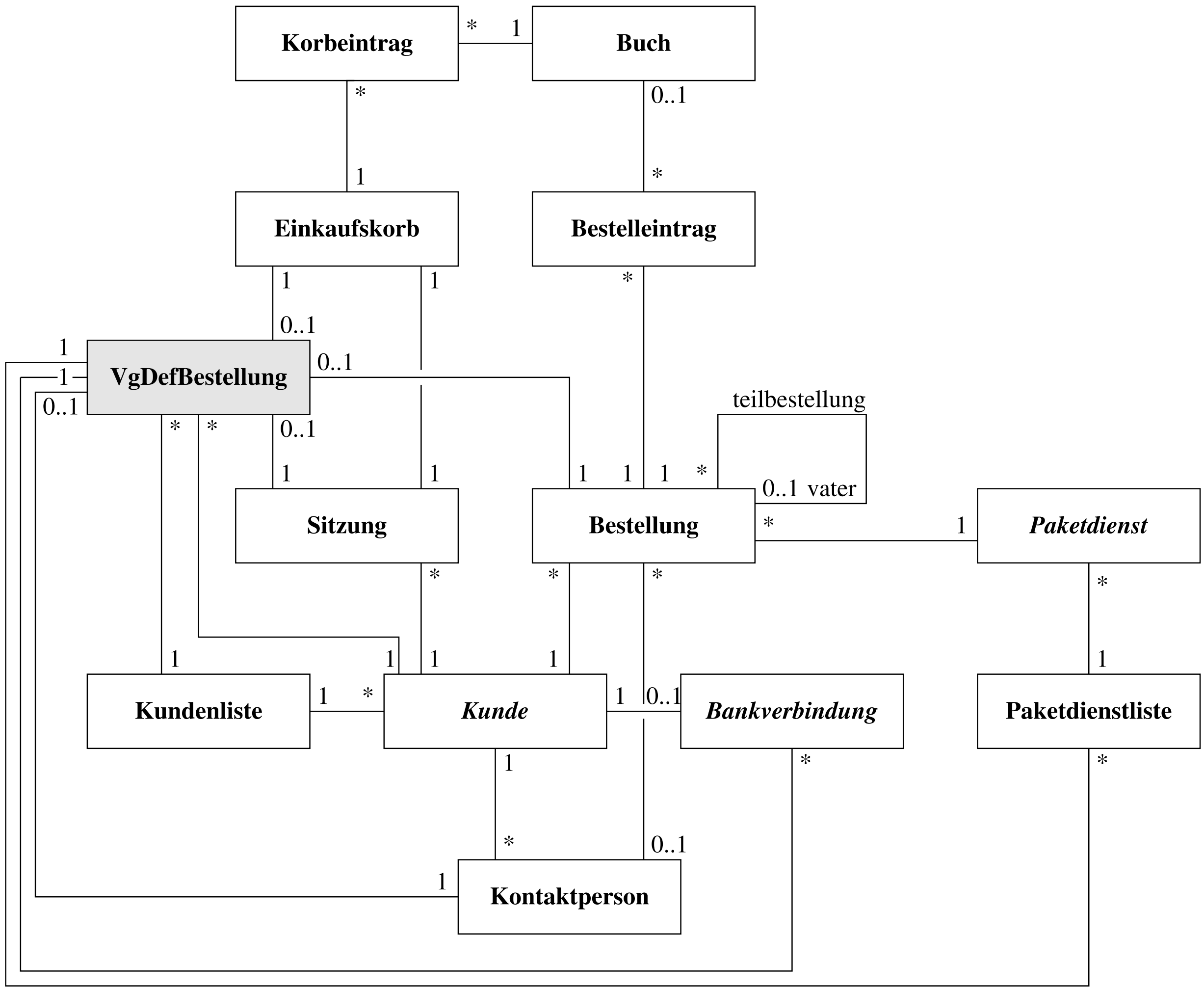


Abbildung 8.2: Klassendiagramm zur Bestellung der im Einkaufskorb liegenden Bücher

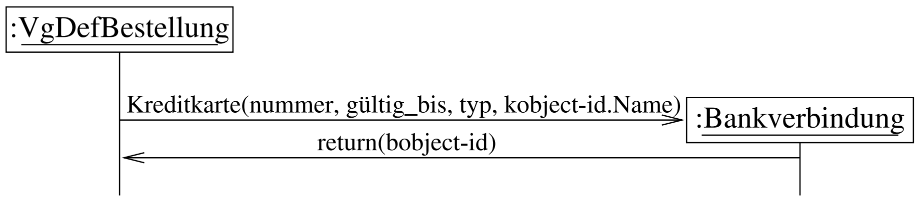


Abbildung 8.3: Ablauf zum Erzeugen eines Kreditkartenobjekts

Im exemplarischen Sequenzdiagramm des Nutzungsfalls „Bestellung der im Einkaufskorb liegenden Bücher“ in den Abbildungen 6.10 auf Seite 94 und 6.11 auf Seite 95 wurde nach der Eingabe einer gewünschten Kreditkartenbezahlung und der dafür notwendigen Daten ein neues Kreditkartenobjekt erzeugt. Anschließend erfolgt für diesen Konstruktor eine Spezifikation der Vor– und Nachbedingung. Abbildung 8.3 zeigt den relevanten Ausschnitt aus dem exemplarischen Sequenzdiagramm.

Die Vorbedingung dient hier zur Überprüfung der Anwendbarkeit der Methode. Die Nachbedingung spezifiziert analog das Ergebnis der Methode, ohne daß über das „Wie“ etwas ausgesagt wird. Die Bedingungen werden in der Sprache OCL beschrieben.

Das Kreditkartenobjekt soll nur erzeugt werden, wenn eine Kreditkartennummer, ein Gültigkeitsdatum, der Typ der Kreditkarte (Visa, Eurocard, ...), und ein Kundenname aus dem Kundenobjekt angegeben wurden. Das Gültigkeitsdatum muß „aktueller“ als das Tagesdatum sein, denn wenn die Lieferung innerhalb von z.B. 24 Stunden erfolgt, wird die Rechnung und Abbuchung am nächsten Tag vollzogen, so daß dann die Kreditkarte ihre Gültigkeit bereits verloren hätte.

Für die folgende Vorbedingung wird vorausgesetzt, daß es im System eine Klasse „Global“ gibt, in der eine Methode das aktuelle Datum liefert. Außerdem soll diese Klasse auch die möglichen Kartentypen in einem mengenwertigen Attribut „paytypes“ halten.

Für die Nachbedingung muß nur die Existenz des neuen Kreditkartenobjekts überprüft werden. Falls das Objekt erzeugt wurde, enthält das aktuelle Objekt eine gültige Referenz auf sich selbst. Die Vor– und Nachbedingung, eingeleitet mit dem Schlüsselwort „pre“ bzw. „post“, sind anschließend aufgelistet.

```
Kreditkarte::Kreditkarte(nummer: Integer, gültig_bis: date, typ: paytype,
  name: String)

  pre:  Global.isValidCardNumber(nummer) and
        gültig_bis <> Global.getDate() and
        Global.paytypes->exists(typ) and
        name.size > 0

  post: Kreditkarte.allInstances->size =
          Kreditkarte.allInstances@pre->size + 1 and
        not Kreditkarte.allInstances@pre->includes(self) and
        Kreditkarte.allInstances->includes(self)
```

Die Anwendung von Vor– und Nachbedingungen sollte hier exemplarisch dargestellt werden.

Im folgenden wird das in diesem Abschnitt erweiterte Klassendiagramm aus Abbildung 8.2 auf der vorherigen Seite nochmals genauer spezifiziert. In diesem Klassendiagramm sollen erstens die ungerichteten Assoziationen in gerichtete transformiert werden und zweitens, wenn möglich, eine Unterscheidung zwischen Aggregation und Komposition getroffen werden, wobei zur Klärung dieser Begriffe auf die Einführung im Abschnitt 8.1 verwiesen sei.

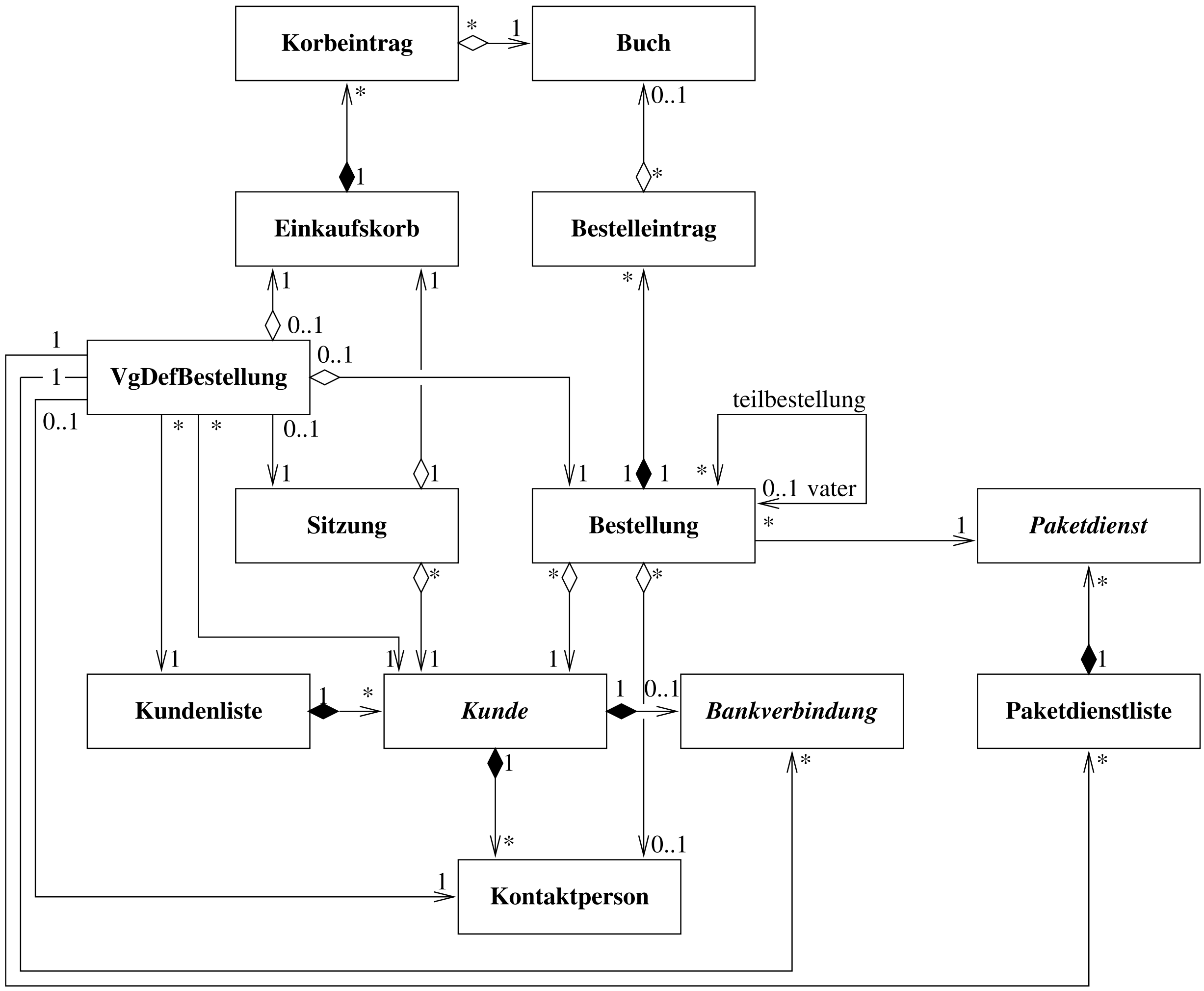


Abbildung 8.4: Klassendiagramm mit gerichteten Assoziationen, Aggregationen und Kompositionen

Verwalten Listenelemente ihre Listeneinträge, so werden die Einträge beim Entfernen der Liste ebenfalls gelöscht, so daß dabei eine Kompositionsbeziehung vorherrscht. Im Diagramm sind einige solcher Listenkompositionen neben anderen Kompositionen zu sehen.

Die Aggregations– und Kompositionsbeziehungen existieren nur auf der Modellierungsebene, d.h. sie werden von keiner Programmiersprache unterstützt. Sie erfaßen wichtiges Wissen aus dem fachlichen Bereich. Außerdem dienen sie bei einer späteren Implementierung zur Klärung über das Löschen und Erzeugen von Objekten, d.h. die hierfür zuständigen Objekte können erkannt und entsprechend implementiert werden.

8.4 Vormerken zum Kauf

Dem Fachklassendiagramm aus Kapitel 4 wird im folgenden auch die Vorgangsklasse zum Vormerken von Büchern, wie sie im Teilabschnitt 6.3 auf Seite 82 entwickelt wurde, mit ihren Assoziationen hinzugefügt. Außerdem werden explizite Listen–Klassen zusätzlich im Diagramm eingefügt, so daß die Assoziationen den genauen Klassen zugeordnet werden können.

Im zu erstellenden Diagramm werden aber nun nicht alle Klassen in weitere Teilklassen unterteilt. Es wird nur an zwei Listenklassen exemplarisch gezeigt. Außerdem muß nicht jede Liste als eigene Klasse im Klassendiagramm dargestellt werden, da für Listen in vielen Programmiersprachen sogenannte „Templates“ zur Verfügung stehen. Die UML bietet hierzu die Möglichkeit, generische Klassen zu definieren (vgl. [FS98]). Der Eintrag einer Liste im Klassendiagramm erscheint jedoch dann sinnvoll, wenn mehrere Klassen Verbindungen zu der Listenklasse herstellen.

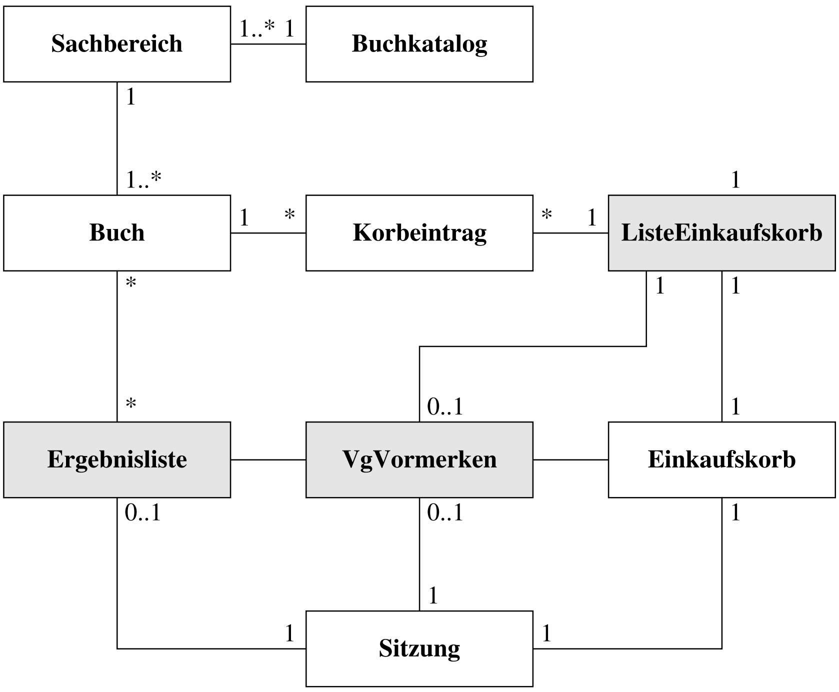


Abbildung 8.5: Klassendiagramm zum Vormerken von Büchern

In Abbildung 8.5 ist das erweiterte Klassendiagramm mit der Vorgangsklasse zum Vormerken von Büchern inklusive der neuen Assoziationen dargestellt. Zur Übersichtlichkeit sind die neu hinzugefügten Klassen wieder grau hinterlegt. Wie im Diagramm zu sehen ist, bestehen Assoziationen zwischen den Listenklassen und der neuen Vorgangsklasse „VgVormerken“. Wäre z.B. die hinzugefügte Klasse „Ergebnisliste“ nicht eingetragen, so würde von der Vorgangsklasse eine Aggregation zur Buchklasse führen. Die Liste der vorzumerkenden Bücher als Ergebnis der Suchanfrage wäre implizit in der Vorgangsklasse enthalten.

Da sich aber im Abschnitt 6.3 auf Seite 82 gezeigt hat, daß die Ergebnisliste auch bei der Büchersuche verwendet wird, war eine explizite Darstellung im Diagramm sinnvoll (siehe hierzu auch Abschnitt 8.5).

Klassendiagramme müssen nicht notwendigerweise bis auf die unterste Detaillierungsebene spezifiziert werden. Je nach Zusammenhang und Referenzierung ist es sinnvoll, weitere Klassen herauszuarbeiten und explizit darzustellen. Dabei ist zu beachten, daß bei einer sehr feinen Struktur die Übersicht über die wesentlichen Klassen verloren gehen kann, obwohl kein entscheidender Informationsgewinn erzielt wird.

8.5 Die Büchersuche

Auch in diesem Abschnitt wird das Fachklassendiagramm von Seite 50 wieder um eine Vorgangsklasse ergänzt. Hierbei kommt auch eine Vererbungshierarchie zum Vorschein.

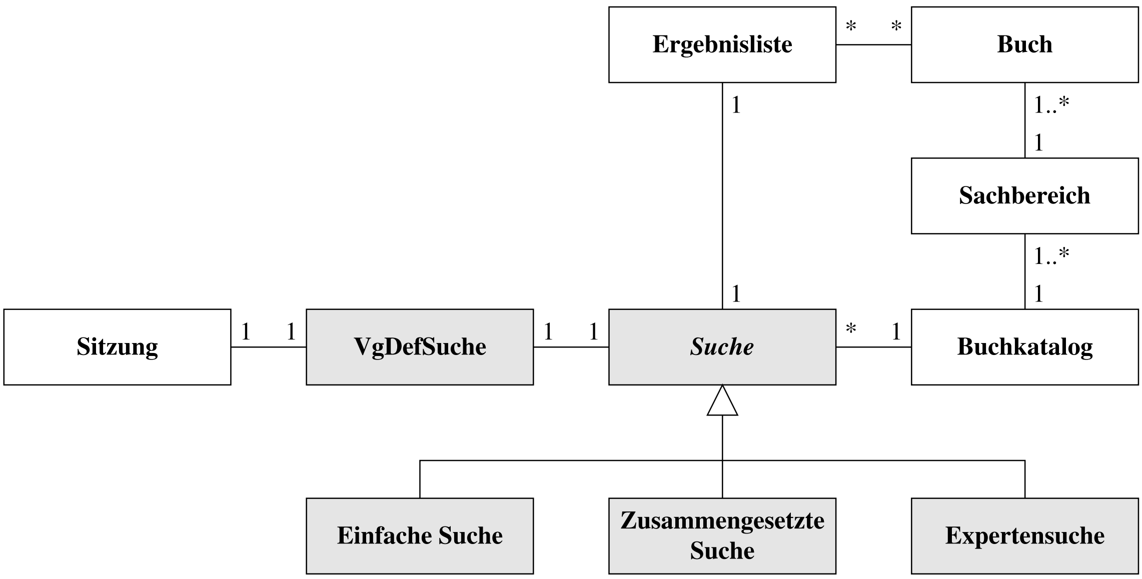


Abbildung 8.6: Klassendiagramm zur Büchersuche

Bei der Spezifikation des Nutzungsfalls zur Büchersuche im Teilabschnitt 6.5 auf Seite 109 wurde bereits die abstrakte Klasse Suche mit ihren Vererbungsbeziehungen zu der einfachen Suche, zur zusammengesetzen Suche und zur Expertensuche angesprochen. Abbildung 8.6 zeigt das erweiterte Klassendiagramm zur Büchersuche, bei dem die neu eingefügten Klassen wieder grau hinterlegt sind. Vererbungsbeziehungen zwischen der Superklasse Suche und ihren Subklassen wurden ebenso wie Aggregationsbeziehungen wieder eingezeichnet. Zu erwähnen ist noch, daß die explizit dargestellte Einkaufsliste bereits bei der Erweiterung des Klassendiagramms zum Vormerken von Büchern (siehe Abschnitt 8.4 auf Seite

140) eingefügt wurde, da das Suchergebnis aus der Büchersuche zum „Büchervormerken“ verwendet werden kann.

8.6 Zusammenfassung

Bei der Analyse der Sitzungsklasse wurde beispielhaft die Ergänzung der Fachklassen mit Attributen und Operationen gezeigt. Da im Fachklassendiagramm aus Kapitel 4 nur die statischen Daten eingebunden wurden, kamen bei der Ergänzung vor allem die für das dynamische Verhalten notwendigen Attribute hinzu. Die Methoden, die das Verhalten inhärent steuern, wurden ebenfalls in das Klassendiagramm mitaufgenommen. Eine solche Analyse und Ergänzung müßte bei einer vollständigen Ausarbeitung an jeder Fachklasse durchgeführt werden. In diesem Kapitel fand eine exemplarische Darstellung hierzu an einem Klassendiagramm statt.

Die Ergänzung des statischen Fachklassendiagramms mit Vorgangsklassen wurde an drei Beispielen gezeigt. Die zusätzlich erforderlichen Assoziationen mußten dabei ebenfalls ergänzt werden. Neben der Erweiterung um reine Vorgangsklassen wurden noch weitere Klassen, die sich bei der Erarbeitung des dynamischen Verhaltens ergaben, in das Klassendiagramm mitaufgenommen. Die hierzu eingefügten Klassen waren Listenklassen (siehe Abbildungen 8.5 auf Seite 141 und 8.6 auf der vorherigen Seite). Die explizite Darstellung von Listenklassen erscheint nur dann sinnvoll, wenn sie von mehreren Fach– bzw. Vorgangsklassen referenziert wird, da ansonsten dadurch meist nur das Klassendiagramm unübersichtlicher wird und keine entscheidende Informationssteigerung mit sich bringt. Außerdem können Klassendiagramme auch in verschiedenen Detaillierungsgraden entwickelt werden. In diesem Zusammenhang sei auf die abschließende Bemerkung im Abschnitt 8.4 hingewiesen. Die Listenklassen könnten auch mit Iteratorklassen ergänzt werden, die ebenfalls in die Klassenstruktur eingetragen werden könnten. Zur Ermittlung von Iteratorklassen könnte z.B. das Iterator–Entwurfsmuster aus [GHJV98] benützt werden.

Die deklarative Spezifikation von Methoden in Form von Vor– und Nachbedingungen wurde exemplarisch dargestellt.

Gerichtete Assoziationen geben Auskunft darüber, in welche Klasse ein Referenzattribut aufgenommen werden muß. Aggregationen und Kompositionen sind eine genauere Spezifizierung der IST–TEIL–VON–Beziehungen, bei denen über die Lebenszeit der assoziierenden Klassen Information im Klassendiagramm dargestellt werden kann.

Listenklassen sind spezielle Managerklassen, wie sie in [Bre98a] vorgestellt werden. Instanzen von Managerklassen verwalten Objektmengen.

Vererbungsbeziehungen traten im Klassendiagramm selten auf. Bei der Klasse Sitzung stand eine Einführung einer Vererbung bis hin zu einer Privatkundensitzung und zu einer Firmenkundensitzung bereits zur Diskussion. Da sich aber die Unterscheidung nur auf sehr kleine Teile der Sitzungsmethoden auswirkt, wurde eine Vererbungsbeziehung zur Sitzungsklasse weggelassen. An Stelle der dadurch entstandenen überschriebenen Methoden müssen bei der Implementierung der Sitzungsmethoden einige zusätzliche Fallunterscheidungen eingeführt werden. Dies bringt jedoch den Nachteil mit sich, daß dieses Wissen auf abstrakter Ebene, hier im Klassendiagramm, nicht sichbar wird. Dieser Nachteil wurde jedoch bei der Sitzungsklasse aus Vereinfachungsgründen in Kauf genommen.

Vererbungsbeziehungen kämen auch bei der Anbindung einer graphischen Oberfläche sehr stark zum Vorschein, da z.B. Fenstereigenschaften durch Klassenvererbungen oder Schnittstellenvererbungen (z.B. abgeleitete Interfaces in der Programmiersprache Java) weitergereicht werden. Die Anbindung an eine graphische Oberfläche wurde hier jedoch nicht behandelt.

Die nochmalige Anwendung von Refactoring–Methoden wäre grundsätzlich nach der Bearbeitung des Klassendiagramms nochmals angebracht. Da aber hier nur die Sitzungsklasse mit Attributen und Methoden ergänzt wurde, und diese Klasse keinen Ansatzpunkt für ein Refactoring liefert, kam es diesbezüglich zu keiner Anwendung in diesem Kapitel.

Kapitel 9

Zusammenfassung und Bewertung

Der in dieser Arbeit durchgeführte objektorientierte Entwurf beginnt mit einem klassischen Pflichtenheft. Ein Pflichtenheft dient neben der Produktbeschreibung als Vertragsgrundlage zwischen dem Auftragsteller und dem Systementwickler. Aus diesem Grund ist es auch weiterhin sinnvoll, auf ein herkömmliches Pflichtenheft aufzusetzen, denn dadurch, daß dieses für eine vertragliche Bindung bei realen Produktentwicklungen angefertigt werden muß, wäre es sinnlos, eine spezielle Form der Produktbeschreibung als Ausgangspunkt für einen objektorientierten Entwurf auszuarbeiten und anzuwenden.

Da ein vollständig ausgearbeitetes Pflichtenheft nach [Bal96] z.B. auch Informationen über Produkt–Schnittstellen enthält, erweist sich dessen Anwendung als nicht geeignet. Bei einer objektorientierten Modellierung werden zu Beginn alle mit dem Produkt in Verbindung stehenden externen Teilsysteme mitmodelliert. Sobald beim Entwurf in seiner Ausarbeitung und Darstellung ein für die Abgrenzungsfestlegung geeignetes Stadium erreicht wird, werden diejenigen Teile, die von anderen Teilsystemen übernommen werden, nicht weiter mitmodelliert. An den abgrenzenden Klassen müssen dann die Schnittstellen spezifiziert und eingehalten werden. Neben einem vollständigen Pflichtenheft existiert noch ein sogenanntes „grobes Pflichtenheft“ (vergleiche [Bal96]), das sich mit solchen Feinheiten wie z.B. Abgrenzungskriterien nicht auseinandersetzt. Diese Form, oftmals auch als Lastenheft bezeichnet, reichte hier für die Spezifizierung des zu entwickelnden Produkts aus und wurde in dieser Arbeit auch angewendet.

Aus den Produktdaten wurde dann im Kapitel 4 das Fachklassendiagramm erstellt. Dieses Diagramm, welches die statische Klassenstruktur der Anwendung repräsentiert, wurde nicht in einem Stück modelliert. Partielle Fachklassendiagramme, die meist aus einer Produktdatenbeschreibung im Pflichtenheft hervorgehen, wurden zu Beginn direkt in entsprechende Klassen übernommen. Anschließend fand, wenn möglich eine Optimierung bezüglich Redundanzfreiheit statt. Wo es sinnvoll war, wurden Ableitungsbeziehungen eingefügt. Bereits in diesem Stadium kamen dann, wenn die erstellte Struktur Anlaß gab, Refactoring–Methoden zum Einsatz. Diese führten im Falle einer Anwendung innerhalb dieser Arbeit zu einer Einführung von Vererbungsbeziehungen und zum Verschieben von Attributen innerhalb eines Vererbungs–Pfades, inklusive der Auswirkungen auf Klassen der selben Hierarchiestufe. Da im Fachklassendiagramm außer den impliziten Attributzugriffs– und Zuweisungs–Methoden noch keine weiteren Operationen spezifiziert sind, wurde ein Refactoring bezüglich der Operationen nicht durchgeführt.

Da zum Refactoring wie auch zu allen anderen Transformations– und Verfeinerungstechniken keine Werkzeuge (Entwicklungs–Tools) zur Verfügung standen, fand diesbezüglich auch keine Anwendung statt, obwohl hier in diesem Stadium geeignete Refactoring–Tools angewendet werden könnten.

Die partiellen Fachklassendiagramme konnten ohne Modifikation zum Gesamt–Fachklassendiagramm vereinigt werden. Da bis auf einige wenige zusätzliche Assoziationen die partiellen Diagramme ohne Änderung übernommen werden konnten, bereitete das Zusammenfügen keine Schwierigkeiten. Weitere Vererbungsbeziehungen kamen dabei nicht zum Vorschein, so daß nach diesem Schritt kein weiteres Refactoring anzuwenden war.

Wie aus dem Zeitplan in Kapitel 2 (siehe Abbildung 2.1 auf Seite 7) hervorgeht, fand die Bearbeitung der Kapitel 4 und 5 parallel statt. Einerseits ergaben sich bei der Bearbeitung der Fachklassen auch bereits bestimmte Abläufe, so daß es sinnvoll erschien, diese in diesem Entwicklungsstadium festzuhalten und als Systemszenarien auszuarbeiten. Zum anderen traten bei der Identifikation der Nutzungsfälle auch die Produktdaten, die im Pflichtenheft nur ungenau spezifiziert waren, hervor. Zwar hätte man das Fachklassendiagramm zuerst entwickeln können, dann eine Identifizierung der Nutzungsfälle abwickeln und in einem geeigneten Kapitel nochmals die Auswirkungen auf das Fachklassendiagramm betrachten können. In diesem Fall wären aber die Auswirkungen so umfangreich gewesen, daß das Fachklassendiagramm völlig neu zu erstellen gewesen wäre, denn besonders die Verbindungen zwischen einzelnen Klassen wurden erst mit der Betrachtung der Systemszenarien klar. Der hier gewählte Ansatz, d.h. die parallele Entwicklung, funktionierte einwandfrei.

Die Analyse der Produktfunktionen im Pflichtenheft und der intuitiven Informationen aus der Fachklassenentwicklung führte zu den Nutzungsfällen auf Systemebene. Zwar wurde hier nur der Nachrichtenaustausch zwischen dem Nutzer und dem ganzen System betrachtet, so daß die Fachklassen hier nicht implizit ins Spiel kamen, aber dennoch war es wichtig, Informationen in Zusammenhang mit der statischen Klassenhierarchie mitzuverwenden. Außer den Nachrichtenflüssen zwischen Nutzer und System wurden teilweise auch wichtige systeminterne Nachrichten ermittelt und dargestellt.

Wie schon in der Zusammenfassung von Kapitel 5 auf Seite 75 erwähnt wurde, eignen sich die Sequenzdiagramme inklusive der Anwendung von Stellvertretern sehr gut zum Beschreiben solcher Nutzungsfälle auf Systemebene. Da der Detaillierungsgrad der Sequenzdiagramme nicht vorgeschrieben ist, kann er, je nach gewünschtem Informationsgehalt, frei gewählt werden, denn bei einigen einfachen Nutzereingaben können diese oftmals zu einer gemeinsamen Nachricht zusammengefaßt werden. Anderenfalls ist es auch auf der Systemebene notwendig, systeminterne Nachrichtenflüsse zum späteren Aufgreifen, z.B. bei der

Bearbeitung der interaktionsorientierten Spezifikation, bereits festzulegen. Als Nachteil bei der Entwicklung der Sequenzdiagramme stellte sich die hohe Anzahl an Teildiagrammen heraus. Da Stellvertreterobjekte wiederum auf einzelne Sequenzdiagramme verweisen, entsteht oftmals für jede Alternative mindestens ein weiteres Diagramm. Bei vielen Varianten in oder zu einem Nutzungsfall sind deshalb sehr viele Diagramme zu erstellen.

Die Betrachtung von systeminternen Nachrichtenflüssen war der zentrale Mittelpunkt des Kapitels 6. Ausgehend von den bereits entwickelten Systemszenarien, dem erstellten Fachklassendiagramm und dem verbalen Pflichtenheft wurden hier die Nutzungsfälle ausgearbeitet. Nach einer verbalen Spezifikation, die den Akteur, die Ein– und Ausgabedaten, die veränderten Fachklassen, eine Ablaufbeschreibung und Alternativbeschreibungen exakt festlegte, konnte ein Nutzungsfalldiagramm erstellt werden. Diese objektorientierten Spezifikationen dienten zur Ausarbeitung von exemplarischen Sequenzdiagrammen und Highlevel Message Sequence Charts (HMSCs). Während in den exemplarischen Sequenzdiagrammen nur beispielhafte Abläufe dargestellt wurden, konnten durch die Verwendung von HMSCs auch Schleifen und Alternativen auf der Ebene der systeminternen Analyse ermittelt werden. HMSCs, die nicht Bestandteil der UML 1.3 sind und aus dem Standard MSC–96 stammen, der besonders bei der Darstellung von Kommunikationsprotokollen verwendet wird, eigneten sich ausgezeichnet zu der Alternativ– und Schleifendarstellung. Da sie auch eine hierarchische Schachtelung erlauben, können Feinheiten abstrahiert und in eigenen HMSCs oder Sequenzdiagrammen betrachtet werden, so daß damit eine sehr übersichtliche Darstellung erlaubt wird. Die Erweiterung der Sequenzdiagramme bezüglich Schleifen und Alternativen eignet sich nur bei kleinen „Schleifenrümpfen“ und „Alternativrümpfen“, da ansonsten leicht die Übersicht verloren gehen kann. Sowohl HMSCs als auch die Erweiterungen zu Sequenzdiagrammen erlauben eine Darstellung paralleler Abläufe. In der Zusammenfassung von Kapitel 6 auf Seite 115 wurden bereits Sequenzdiagramme und HMSCs gegeneinander abgewogen.

Zur Verhaltensbeschreibung einzelner Objekte wurden Statecharts angewendet. Dabei wurden die exemplarischen Sequenzdiagramme mit Objektzuständen angereichert und in Automaten transformiert. Durch die unterschiedliche Einführung von Zuständen in Anzahl und Abstraktionsstufe können unterschiedliche Detaillierungsgrade in den Automaten erreicht werden. Auch die Verhaltensunterschiede einer Super– und Subklasse können durch Zustandsverfeinerungen dargestellt werden. Die Statecharts eignen sich sehr gut für die Verhaltensbeschreibung von einzelnen Objekten. Das Verhalten konnte gegenüber der Beschreibung mit HMSCs und Sequenzdiagrammen noch weiter spezifiziert werden. Z.B. konnte das Abbruchverhalten im Statechart ergänzt werden. Eine Automatenverfeinerung entlang der Klassenhierarchie (vgl. Abschnitt 7.5 auf Seite 133) erlaubt auch die Verhaltensbeschreibung einer abgeleiteten Klasse, wobei die Unterschiede durch die zusätzlich eingeführten Zustände sehr deutlich zum Vorschein kommen. Beim Vergleich der zugehörigen Statecharts können die Verhaltensunterschiede ohne weitere Zusatzinformation, z.B. aus anderen Diagrammen, ermittelt werden.

Der schrittweise Ansatz zur Transformation von exemplarischen Sequenzdiagrammen zu einem Statechart bietet die Möglichkeit, auf einheitliche Ergebnisse nach der Transformation zu stoßen. Sobald sich dieses Vorgehen als praktisch anwendbar erweist, können auch Werkzeuge erstellt werden, die die Transformation geeignet unterstützen.

Neben der Verhaltensbeschreibung von ganzen Objekten können Statecharts zur Zustandsänderung bezüglich den möglichen Wertbelegungen von Statusvariablen erstellt werden. Hierdurch kann die Vollständigkeit der Transitionen ermittelt werden, denn für jeden Zustandswechsel müssen in den Nutzungsfällen Nachrichtenflüsse vorliegen. Da jedoch im Rahmen dieser Arbeit nicht alle Nutzungsfälle ausgearbeitet wurden, konnten auch nicht alle Verbindungen zu Transitionen ermittelt werden (vergleiche hierzu die Zusammenfassung zu Kapitel 7 auf Seite 133).

Im Kapitel 8 wurde abschließend noch einmal die Klassenstruktur betrachtet. Vorgangsklassen, die das Verhalten eines Nutzungsfalls festlegen, wurden in das Fachklassendiagramm mit den notwendigen Assoziationen eingebracht. Wie in diesem Kapitel festgestellt wurde, ist es sinnlos, alle möglichen Klassen in ein Klassendiagramm einzuzeichnen, da ansonsten die Übersichtlichkeit der Diagramme verloren geht. Listenklassen, die z.B. nur von einer einzigen Klasse referenziert werden, liefern keinen entscheidenden Informationsgewinn.

Beispielhaft wurde auch die Ergänzung von Attributen und Methoden an einer Fachklasse gezeigt. Eine nochmalige Anwendung von Verfeinerungstechniken an den so analysierten Fach– und Vorgangsklassen wäre in diesem Stadium möglich. Da jedoch im Rahmen dieser Arbeit Methoden– und Attributergänzungen nur an einer Klasse gezeigt wurden und sich dabei herausstellte, daß an dieser Klasse keine Verfeinerung notwendig ist, kam hier eine nochmalige Verfeinerung nicht zum Einsatz. Andere Klassen für eine Verfeinerung bezüglich Attributen und Operationen waren ohne weitere Analyse nicht ersichtlich.

Exemplarisch wurde auch die Anwendung von Invarianten, Vor– und Nachbedingungen gezeigt. Hierzu wird auf die Zusammenfassung von Kapitel 8 auf Seite 143 verwiesen. Die Sprache OCL, die sowohl in Kapitel 4 zur Beschreibung von Invarianten als auch in Kapitel 8 für Vor– und Nachbedingungen angewendet wurde, besitzt für die meisten Anwendungsfälle die nötige Aussagekraft. Ein Ausdruck z.B. zur Bestimmung des kleinsten Fixpunktes war jedoch mit ihr nicht möglich. Die Kombination aus textueller OCL und den graphischen Beschreibungstechniken hat sich in dieser Arbeit sehr gut bewährt. Die OCL ergänzte Designentscheidungen, die durch eine reine Anwendung von Diagrammen nicht möglich gewesen wäre.

Der in dieser Arbeit verwendete nutzungsfallorientierte Entwicklungsprozeß eignete sich sehr gut für die Bearbeitung dieses Fallbeispiels. In den einzelnen Entwicklungsschritten wurden ausgehend vom Pflichtenheft die nötigen Klassendiagramme, Sequenzdiagramme und Statecharts neben HMSCs und Nutzungsfalldiagrammen erstellt. Die schrittweise Bearbeitung lieferte immer genügend Information für die weiteren Entwurfsschritte, so daß mit Ausnahme der parallelen Bearbeitung von Fachklassendiagramm und Systemszenarien eine auf die vorausgehenden Entwicklungsergebnisse aufbauende Entwicklung vollzogen werden konnte.

Der derzeitige Entwicklungsstand der UML 1.3 eignet sich in weiten Teilen für die Ausarbeitung der notwendigen Diagramme. Jedoch sind Techniken zur Transformation von UML–Modellen zum Zweck der Verfeinerung und Komposition, wie sie auch teilweise schon entwickelt und in dieser Arbeit verwendet wurden, als Bestandteil der UML zu einer vollständigen Bearbeitung notwendig. Da die Entwicklung der UML noch nicht abgeschlossen ist werden sich in den nächsten Jahren diese Lücken schließen.

Vollständige Beschreibungs– und Transformationstechniken zur Modellierung sämtlicher Softwareprodukte wird es jedoch in absehbarer Zukunft nicht geben, da ständig neue Richtungen bei der Softwareentwicklung eingeschlagen werden. Im Vergleich z.B. zum Maschinenbau ist das Gebiet der Informatik noch zu neu und zu unerforscht, daß hier allgemeingültige und abschließende Entwurfsmethoden genannt werden können. Auch die steigende Komplexität von Software läßt einen Entwurfsstandard nicht zu.

Literaturverzeichnis

[Alh98] Sinan Si Alhir. *UML in a Nutshell.* O'Reilly, first edition, 1998.

[Bal96] Helmut Balzert. *Lehrbuch der Software–Technik: Software–Entwicklung.* Spektrum Akademischer Verlag, Erste Auflage, 1996.

[BBH+99] Ruth Breu, Manfred Broy, Franz Huber, Ingolf Krüger, Bernhard Rumpe, and Wolfgang Schwerin. Applied Software Engineering Principles for UML. Book in preparation, Mai 1999.

[Bey98] Hans-Bernhard Beykirch. Ecommerce–Protokoll: Weltweit handeln. OTP: Open Trade Protokoll. In *iX – Magazin für Professionelle Informationstechnik*, Seiten 122–126. Verlag Heinz Heise, März 1998.

[BGH+97a] Ruth Breu, Radu Grosu, Christoph Hofmann, Franz Huber, Ingolf Krüger, Bernhard Rumpe, Monika Schmidt, and Wolfgang Schwerin. Exemplary and Complete Object Interaction Descriptions. In Haim Kilov, Bernhard Rumpe, and Ian Simmonds, editors, *Proceedings OOPSLA'97 Workshop on Object-oriented Behavioral Semantics.* TUM-I9737, 1997. http://www4.informatik.tu-muenchen.de/papers/oopsla_schwerin_1997_Conference.html.

[BGH+97b] Ruth Breu, Radu Grosu, Franz Huber, Bernhard Rumpe, and Wolfgang Schwerin. Towards a Precise Semantics for Object–Oriented Modeling Techniques. In Jan Bosch and Stuart Mitchell, editors, *Object–Oriented Technology, ECOOP'97 Workshop Reader.* Springer Verlag, :NCS 1357, 1997. http://www4.informatik.tu-muenchen.de/papers/ECOOPWS97a.html.

[BGH+98] Ruth Breu, Radu Grosu, Franz Huber, Bernhard Rumpe, and Wolfgang Schwerin. Systems, Views and Models of UML. In Martin Schader and Axel Korthaus, editors, *The Unified Modeling Language, Technical Aspects and Applications*, pages 93–109. Physica Verlag, Heidelberg, 1998. http://www4.informatik.tu-muenchen.de/papers/BGHRS98.html.

[BHH+97] Ruth Breu, Ursula Hinkel, Christoph Hofmann, Cornel Klein, Barbara Paech, Bernhard Rumpe, and Veronika Thurner. Towards a Formalization of the Unified Modeling Language. Technical Report TUM-I9726, Technische Universität München, 1997. http://www4.informatik.tu-muenchen.de/reports/TUM-I9726.html.

[BHP+98] Manfred Broy, Franz Huber, Barbara Paech, Bernhard Rumpe, and Katharina Spies. Software and System Modeling Based on a Unified Formal Semantics. In Manfred Broy and Bernhard Rumpe, editors, *Requirements Targeting Software and Systems Engineering, International Workshop RTSE'97, LNCS 1526*. Springer, 1998. http://www4.informatik.tu-muenchen.de/papers/BHPRS98.html.

[BK+96] Anne Brüggemann-Klein et al. *Project MeDoc: Pflichtenheft*, Version 1.0, Juni 1996. http://www11.informatik.tu-muenchen.de/proj/Medoc1/Pflichtenheft/.

[Boo94] Grady Booch. *Object–Oriented Analysis and Design with Applications*. Addison Wesley Longman Inc., second edition, 1994.

[Bre98a] Ruth Breu. *Konzepte, Techniken und Methodik des objektorientierten Entwurfs – Ein integrierter Ansatz*. Habilitationsschrift, Technische Universität München, November 1998.

[Bre98b] Ruth Breu. *Vorlesungsskript zur Vorlesung Objektorientierte Entwurfsmethoden im Wintersemester 1998/99*. Technische Universität München, 1998.

[BRJ98] Grady Booch, James Rumbaugh, and Ivar Jacobson. *The Unified Modeling Language User Guide*. Addison Wesley Longman Inc., first edition, 1998.

[Bro92] Manfred Broy. *Informatik. Eine grundlegende Einführung. Teil I: Problemnahe Programmierung*. Springer–Verlag, Erste Auflage, 1992.

[BRS97] Klaus Bergner, Andreas Rausch, and Marc Sihling. *Using UML for Modeling a Distributed Java Application*. Technical Report TUM–I9735, Technische Universität München, 1997. http://www4.informatik.tu-muenchen.de/reports/-TUM–I9735.html.

[Bur97] Rainer Burkhardt. *UML - Unified Modeling Language: Objektorientierte Modellierung für die Praxis*. Addison Wesley Longman Inc., Erste Auflage, 1997.

[CAB+94] Derek Coleman, Patrik Arnold, Stephanie Bodoff, Chris Dollin, Helena Gilchrist, Fiona Hayes, and Paul Jeremaes. *Object–Oriented Development: The Fusion Method*. Prentice–Hall, 1994.

[CRF98] Sabine Crede, Bernhard Röttgerkamp und Nicole Forster. *E–Markets: Elektronische Märkte verändern die Geschäftswelt.* Deutsche Telekom Systemlösungen GmBH, Erste Auflage, Dezember 1998.

[DD97] Stephan Dresen und Thomas Dunne. Elektronisches Geld: Penny Lane. Wie das ecash–Projekt der Deutschen Bank praktisch funktioniert. In *iX – Magazin für Professionelle Informationstechnik*, Seiten 102–107. Verlag Heinz Heise, Dezember 1997.

[DD98] Stephan Dresen und Thomas Dunne. Elektronisches Geld: Fürs Netz geprägt. Wie CyberCash funktioniert. In *iX – Magazin für Professionelle Informationstechnik*, Seiten 110–116. Verlag Heinz Heise, April 1998.

[Dre98] Stephan Dresen. Elektronisches Geld: Abgewogen. ecash, Cybercash und Millicent im Vergleich. In *iX – Magazin für Professionelle Informationstechnik*, Seiten 96–98. Verlag Heinz Heise, Mai 1998.

[EBF+98] Andy Evans, J-M. Bruel, Robert France, Kevin Lano, and Bernhard Rumpe. Making UML Precise. In *OOPSLA'98 Workshop on "Formalizing UML. Why and How?" 10 Seiten, Vancouver, Canada. October 1998.* OOPSLA'98, 1998. http://www4.informatik.tu-muenchen.de/papers/EbFlR98o.html.

[EFLR98] Andy Evans, Robert France, Kevin Lano, and Bernhard Rumpe. Developing the UML as a Formal Modelling Notation. In Pierre-Alain Muller and Jean Bezivin, editors, *¡¡UML¿¿'98 Beyond the notation. International Workshop Mulhouse France.* Ecole Superieure Mulhouse, Universite de Haute-Alsace, 1998. http://www4.informatik.tu-muenchen.de/papers/EFLR98b.html.

[EFLR99] Andy Evans, Robert France, Kevin Lano, and Bernhard Rumpe. The UML as a Formal Modeling Notation. In Jean Bezivin and Pierre-Alain Muller, editors, *The Unified Modeling Language – Workshop UML'98: Beyond the Notation.* Springer Verlag Berlin, LNCS, 1999. http://www4.informatik.tu-muenchen.de/papers/EFLR98d.html.

[FELR98] Robert France, Andy Evans, Kevin Lano, and Bernhard Rumpe. The UML as a formal modeling notation (here is the almost final version). In *Computer Standards & Interfaces 19*, pages 325–334, 1998.

[FS98] Martin Fowler and Kendal Scott. *UML Distilled – Applying the Standard Object Modelling Language.* Addison Wesley Longman Inc., 7th edition, June 1998.

[GHJV98] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: elements of reusable object–oriented software.* Addison Wesley Longman Inc., 15th edition, 1998.

[Jac93] Ivar Jacobson. *Object–Oriented Software Engineering – A Use Case Driven Approach.* Addison–Wesley Publishing Company, 4th edition, 1993.

[JBR98] Ivar Jacobson, Grady Booch, and James Rumbaugh. *The Unified Software Development Process.* Addison Wesley Longman Inc., first edition, 1998.

[KGSB99] Ingolf Krüger, Radu Grosu, Peter Scholz, and Manfred Broy. From MSCs to Statecharts. In *Proceedings of DIPES.* Kluwer, 1999.

[KPR97] Cornel Klein, Christian Prehofer, and Bernhard Rumpe. Feature Specification and Refinement with State Transition Diagrams. In P. Dini, editor, *Fourth IEEE Workshop on Feature Interactions in Telecommunications Networks and Distributed Systems.* IOS–Press, 1997. http://www4.informatik.tu-muenchen.de/papers/fiw-std_prehofer_1997_Publication.html.

[KR97] Haim Kilov and Bernhard Rumpe. Summary of ECOOP'97 Workshop on Precise Semantics of Object–Oriented Modeling Techniques. In J. Bosch and S. Mitchell, editors, *Object–Oriented Technology – ECOOP'97 Workshop Reader.* Springer Verlag Berlin, LNCS 1357, 1997. http://www4.informatik.tu-muenchen.de/papers/KR97c.html.

[Krü99] Ingolf Krüger. Distributed System Design with Message Sequence Charts. Dissertation in preparation, Technische Universität München, 1999.

[KRB96] Cornel Klein, Bernhard Rumpe, and Manfred Broy. A stream–based mathematical model for distributed information processing systems – SysLab system model –. ENST France Telecom, 1996. http://www4.informatik.tu-muenchen.de/papers/KleinRumpeBroy_FMfO1996.html.

[KRS97] Haim Kilov, Bernhard Rumpe, and Ian Simmonds. Object–Oriented Behavioral Semantics – With an Emphasis on Semantics of Large OO Business Specifications. In *OOPSLA'97 Conference Addendum to the Proceedings.* ACM press, 1997. http://www4.informatik.tu-muenchen.de/papers/KRS98.html.

[Kru99] Philippe Kruchten. *The Rational Unified Process.* Addison Wesley Longman Inc., first edition, 1999.

[KWC98] Anneke Kleppe, Jos Warmer, and Steve Cook. Informal formality? – The Object Constraint Langugage and its application in the UML metamodel. In *UML'98 International Workshop ESSAIM*, pages 127–135, 1998.

[Lan97] Barbara Lange. Electronic Commerce: Blitzableiter. Secure Electronic Transaction. Kreditkarten im Internet. In *iX – Magazin für Professionelle Informationstechnik*, Seiten 120–124. Verlag Heinz Heise, Oktober 1997.

[Lan98] Barbara Lange. Electronic Commerce: Mausklick–Preise. Abrechnung von Kleinstbeträgen im Internet. In *iX – Magazin für Professionelle Informationstechnik*, Seiten 119–121. Verlag Heinz Heise, Januar 1998.

[Mic99] Delf Michel. Mehr Information, mehr Umsatz. EDV–Buchversand Michel: Bücher und Software übers Web kaufen (Fallstudie). In *e.commerce Magazin*, 1. Ausgabe, Seiten 30, 31. IWT Magazin Verlags–GmbH, März/April 1999.

[Neu98] Horst Neumann. *Objektorientierte Entwicklung mit der Unified Modeling Language (UML)*. Hanser Verlag, Erste Auflage, 1998.

[Oes97] Bernd Oestereich. *Objektorientierte Softwareentwicklung: Analyse und Design mit der Unified Modeling Language*. R. Oldenbourg, Vierte Auflage, 1997.

[OJ93] William Opdyke and Ralph Jonson. Creating Abstract Superclasses By Refactoring. In *Proceedings of CSC´93: 1993 ACM Computer Science Conference*, Indianapolis, Indiana, February 1993. ftp://st.cs.uiuc.edu/pub/papers/refactoring/refactoring–superclass.ps.

[Opd92] William Opdyke. *Refactoring Object–Oriented Frameworks*. PhD thesis, University of Illinois at Urbana–Champaign, Technical Report No. UIUCDCS–R–92–1759, 1992. ftp://st.cs.uiuc.edu/pub/papers/refactoring/opdyke–thesis.-ps.Z.

[PB96] Gustav Pomberger und Günther Blaschek. *Software–Engineering: Prototyping und objektorientierte Software–Entwicklung*. Carl Hanser Verlag, Zweite Auflage, 1996.

[PR97] Barbara Paech and Bernhard Rumpe. Towards Development of Correct Software using Views. In A. Evans and K. Lano, editors, *Proceedings of BCS FACS / EROS ROOM Workshop*. ROOS Project Report GR/K67311-2, 1997. http://www4.informatik.tu-muenchen.de/papers/PaeR97b.html.

[Rat97] Rational. *Unified Modelling Language*, Version 1.1 , September 1997. http://-www.rational.com/uml.

[RBP+91] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, and William Lorenzen. *Object–Oriented Modeling and Design*. Prentice–Hall, 1991.

[RJB98] James Rumbaugh, Ivar Jacobson, and Grady Booch. *The Unified Modeling Language Reference Manual*. Addison Wesley Longman Inc., first edition, 1998.

[Rum96] Bernhard Rumpe. *Formale Methodik des Entwurfs verteilter objektorientierter Systeme*. Herbert Utz Verlag Wissenschaft, Dissertation, Technische Universität München, Erste Auflage, 1996.

[Rum98a] Bernhard Rumpe. A Note on Semantics (with an Emphasis on UML). In Haim Kilov and Bernhard Rumpe, editors, *Second ECOOP Workshop on Precise Behavioral Semantics*. Technische Universität München, TUM-I9813, 1998. http://www4.informatik.tu-muenchen.de/papers/RUM98a.html.

[Rum98b] Bernhard Rumpe. Formale methodik des entwurfs verteilter objektorientierter systeme. In *Ausgezeichnete Informatikdissertationen 1997*. B. G. Teubner Stuttgart, 1998. http://www4.informatik.tu-muenchen.de/papers/Rum98c.html.

[Rum98c] Bernhard Rumpe. *PowerPoint–Folien zu OCL Issues*. Institut für Informatik, Technische Universität München, November 1998.

Abbildungsverzeichnis