

Versandhandel

Hausarbeit in Objektorientierte Analyse und Design
FH Osnabrück
Sommersemester 2004

Christian Dommasch
Matrikelnr: 228552
Email: cdommasc@uos.de

David Duhme
Matrikelnr: 227807
Email: dduhme@t-online.de

Inhalt

1. Anforderungsanalyse
 - 1.1 Formulierung des allgemeinen Ziels
 - 1.2 Auflisten der Bediener und Nutzer des neuen Systems
 - 1.3 Auflisten der wichtigsten Funktionen
 - 1.4 Auflisten besonderer Systemeigenschaften
 - 1.5 Kurzbeschreibung der Anwendungsfälle in Steckbriefform
 - 1.6 Aktivitätsdiagramm je Anwendungsfall

2. Analyse
 - 2.1 Geschäftsklassendiagramm
 - 2.2 Zustandsdiagramme

3. Design
 - 3.1 Sequenzdiagramme
 - 3.2 Systemklassendiagramme
 - 3.3 Erläuterungen zu verwendeten Entwurfsmustern

4. Implementierung

1 Anforderungsanalyse

1.1 Formulierung des allgemeinen Ziels

Erstellt werden soll ein System zur Abwicklung der Prozesse eines Versandhandelsunternehmens.

Im Brennpunkt stehen dabei die Prozesse von Einkauf und Verkauf sowie die "Schnittstelle" zum Kunden. Es werden keine Anforderungen der Finanzbuchhaltung behandelt, um den Umfang der Arbeit nicht zu groß werden zu lassen. Module für den beim Versandhandel zwangsweise stattfindenden Zahlungsverkehr werden also hier nicht implementiert.

Das System soll die Prozesse implementieren, die nötig sind, um den vollständigen Weg von beliebigen Artikeln des Warensortiments abzubilden - vom Einkauf über die Lagerhaltung bis hin zum Versand zum Kunden. Für Endkunden soll ein Frontend erstellt werden, das die Bestellung von Waren mit beliebigen PCs mit Internetanschluss ermöglicht.

Für versendete Waren werden Rechnungen generiert. Die einzig mögliche Zahlungsart für Endkunden soll bis auf weiteres das Bezahlen per Rechnung sein.

1.2 Bediener und Nutzer des neuen Systems

Einkauf, Verkauf, Kunde

1.3 Auflisten der wichtigsten Funktionen

- F1: Artikel neu anlegen
- F2: Artikel bestellen
- F3: Wareneingang bestätigen
- F4: Artikelliste anzeigen
- F5: Artikelauswahl bestellen
- F6: Rechnung erstellen
- F7: Artikel versenden

1.4 Auflisten besonderer Systemeigenschaften

Die Software ist ausschließlich in Java implementiert (Java SDK 1.4.2).

Zur persistenten Datenhaltung wird das Datenbank-Management-System Postgresql eingesetzt (getestet: Posgresql 7.3.4). Transaktionssicherheit wurde dabei nicht sichergestellt, kann aber wohl nachimplementiert werden.

Durch die Verwendung eines Drei-Schichten-Modells können verschiedenste Benutzeroberflächen realisiert werden (Textoberfläche wurde ansatzweise implementiert). Auch die Datenspeicherung per DBMS kann durch diese Architektur relativ einfach ersetzt werden.

1.5 Kurzbeschreibung der Anwendungsfälle in Steckbriefform

Steckbrief F1

Name: Artikel neu anlegen
Akteure: Einkauf
Kurzbeschreibung: Der Einkauf legt im System einen neuen Artikel an, der anschließend beim Lieferanten bestellt werden kann.
Vorbedingung: Ein neuer Artikel soll ins Sortiment aufgenommen werden.
Nachbedingung: Artikelspezifikation liegt im System vor.

Steckbrief F2

Name: Artikel bestellen
Akteure: Einkauf
Kurzbeschreibung: Der Einkauf bestellt bestimmte Stückzahlen von Artikeln beim Lieferanten.
Vorbedingung: Die Artikelspezifikationen befinden sich im System.
Nachbedingung: Ein Bestellauftrag für die bestellten Artikel wird generiert.

Steckbrief F3

Name: Wareneingang bestätigen
Akteure: Einkauf
Kurzbeschreibung: Der Lieferant liefert alle Artikel eines Bestellauftrags. Der Einkauf bestätigt den Wareneingang für diesen Auftrag.
Vorbedingung: Lieferant liefert Artikel.
Nachbedingung: Stückzahlen der Artikel werden im System angepasst. Der zugehörige Auftrag wird im System als "erledigt" markiert.

Steckbrief F4

Name: Artikelliste anzeigen
Akteure: Einkauf, Verkauf, Kunde
Kurzbeschreibung: Im System vorhandene Artikel werden unter bestimmten Such- oder Ordnungskriterien angezeigt.
Vorbedingung: keine
Nachbedingung: Artikel werden angezeigt.

Steckbrief F5

Name: Artikelauswahl bestellen
Akteure: Kunde, Verkauf
Kurzbeschreibung: Ein Kunde bestellt aus dem Sortiment einen oder mehrere Artikel.
Vorbedingung: keine
Nachbedingung: Ein Auftrag wird erstellt und liegt im System vor. Die Artikelstückzahlen werden entsprechend reduziert.

Steckbrief F6

Name: Rechnung erstellen
Akteure: Verkauf
Kurzbeschreibung: Für einen Kundenauftrag wird eine Rechnung erstellt.
Vorbedingung: Kundenauftrag liegt vor.
Nachbedingung: Rechnung wurde erstellt.

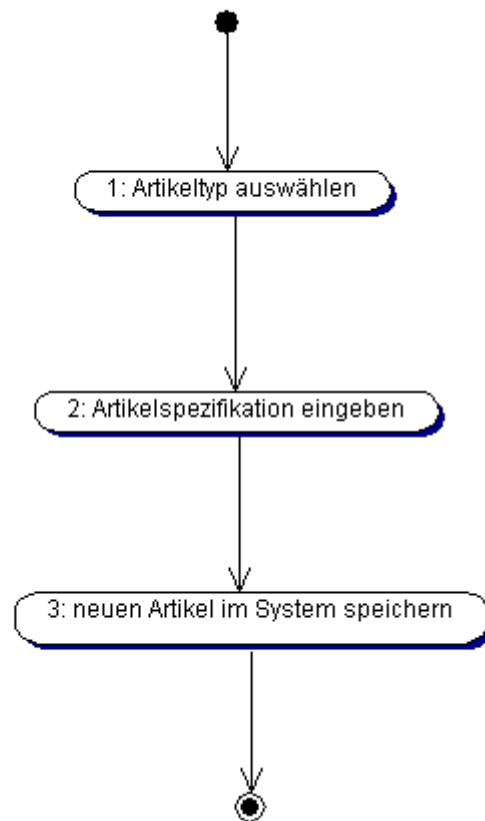
Steckbrief F7

Name: Artikel versenden
Akteure: Verkauf
Kurzbeschreibung: Alle Artikel eines Kundenauftrags werden (zusammen mit der zugehörigen Rechnung) an den Kunden versendet.
Vorbedingung: Auftrag und Rechnung liegen vor. Alle Artikel sind vorrätig.
Nachbedingung: (Buchhaltung wird über den Geschäftsvorgang informiert.)

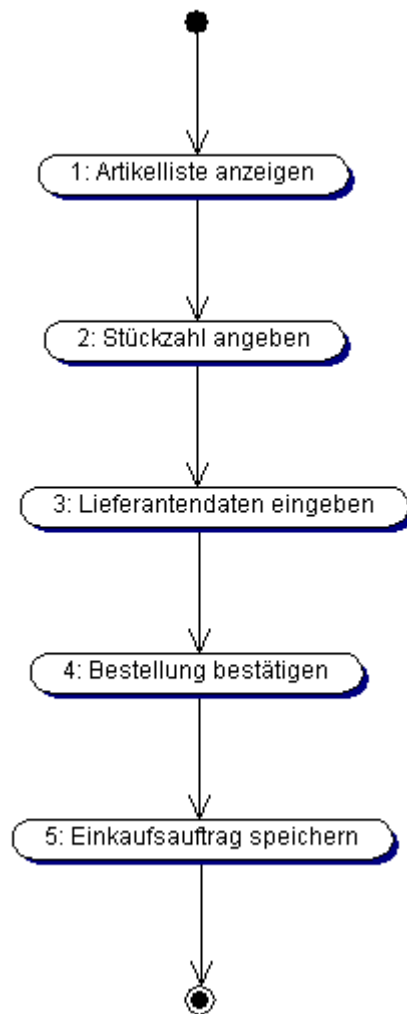
1.6 Aktivitätsdiagramme

Die Anwendungsfälle entsprechen den in 1.4 festgestellten Systemfunktionen.

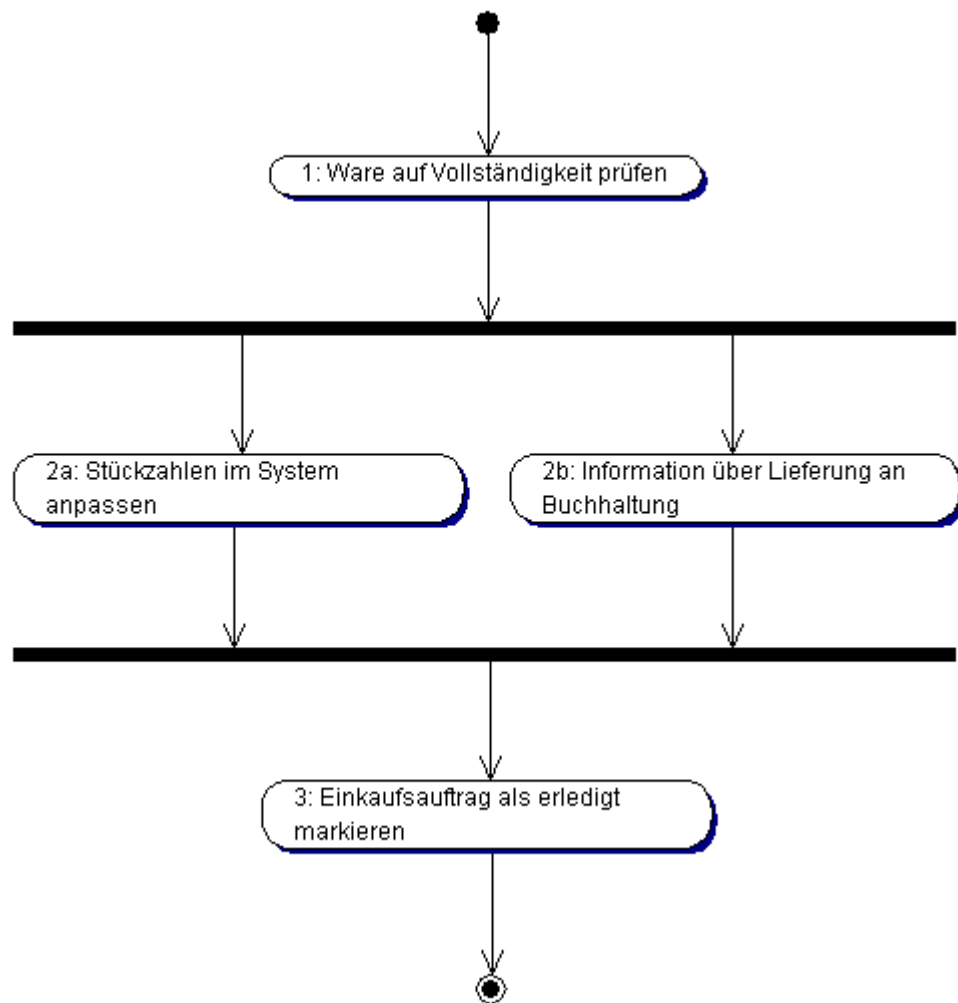
AF 10: Artikel neu anlegen



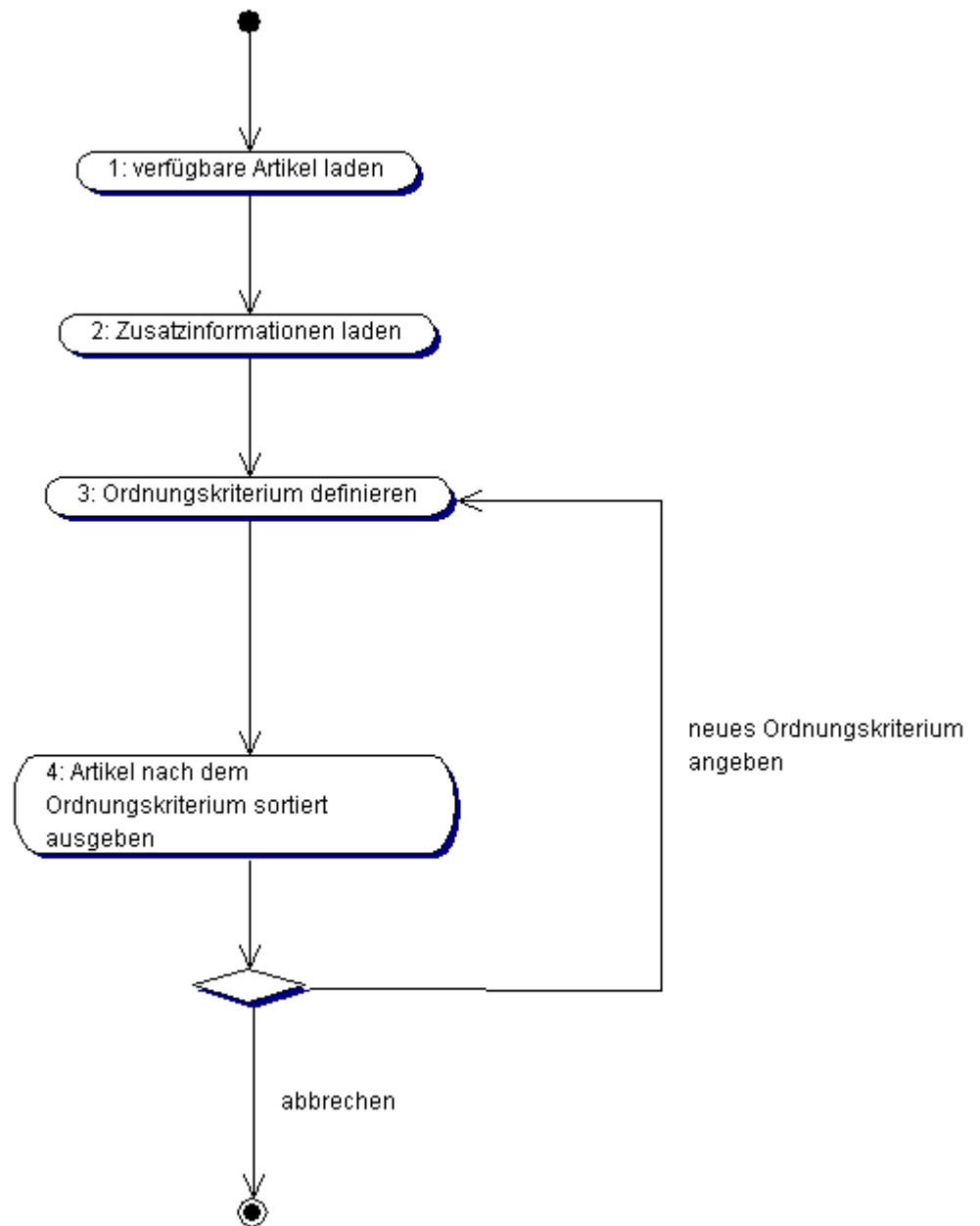
AF 20: Einkaufsauftrag erstellen



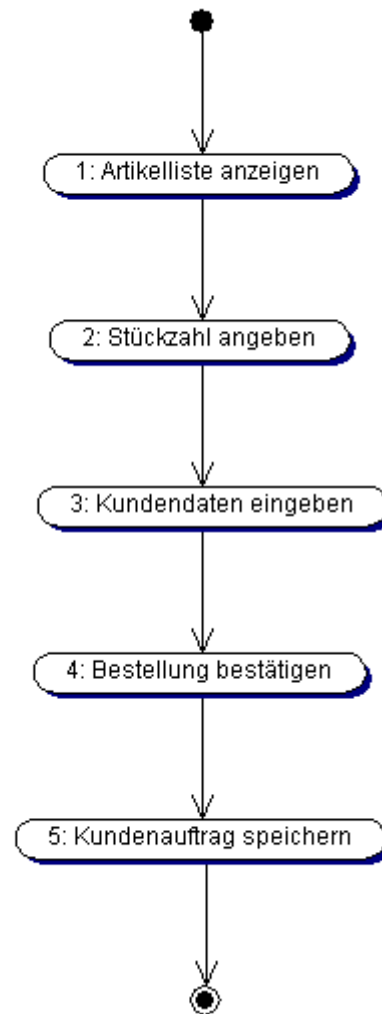
AF 30: Wareneingang bestätigen



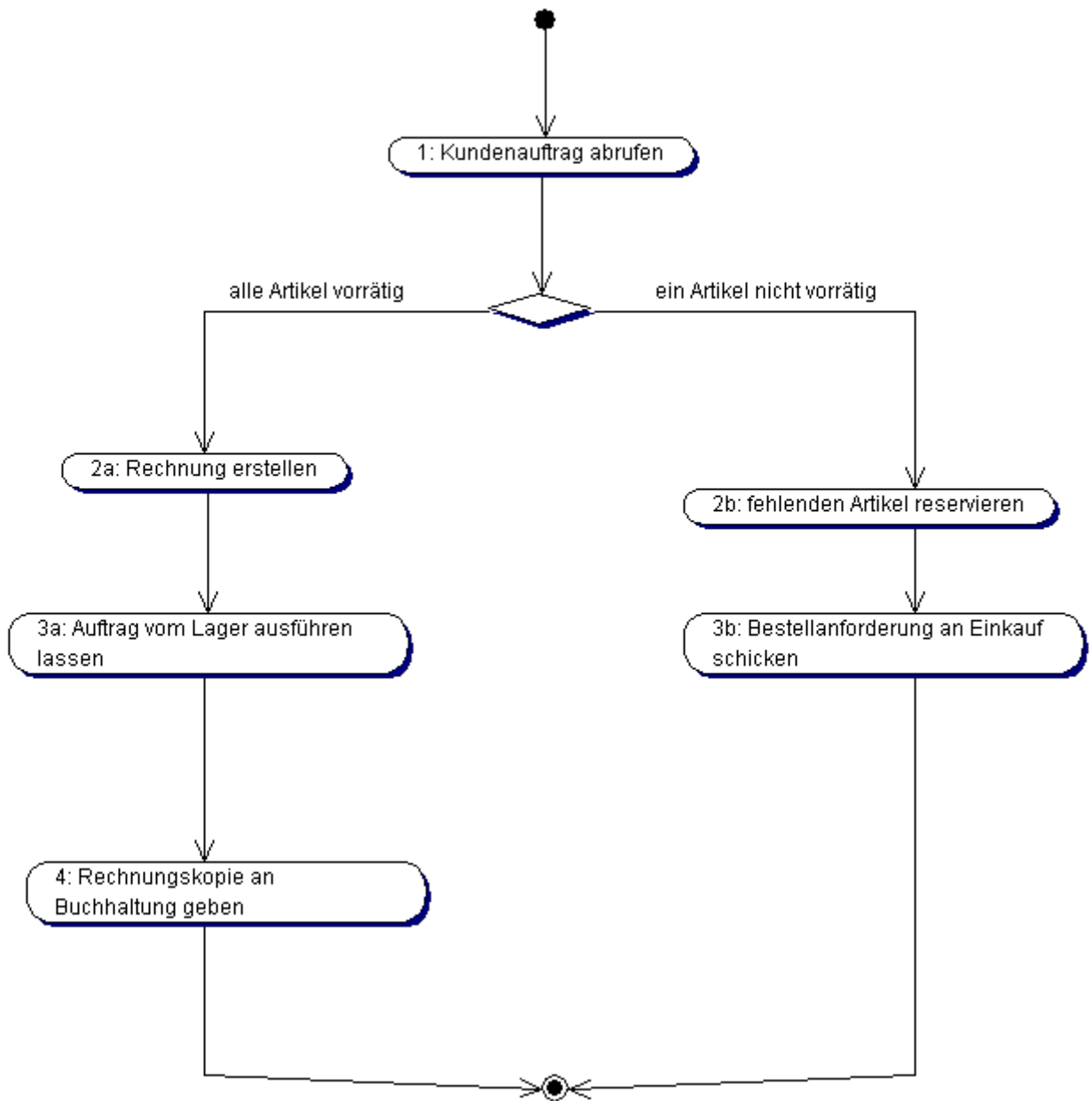
AF 40: Artikelliste anzeigen



AF 50: Kundenauftrag erstellen

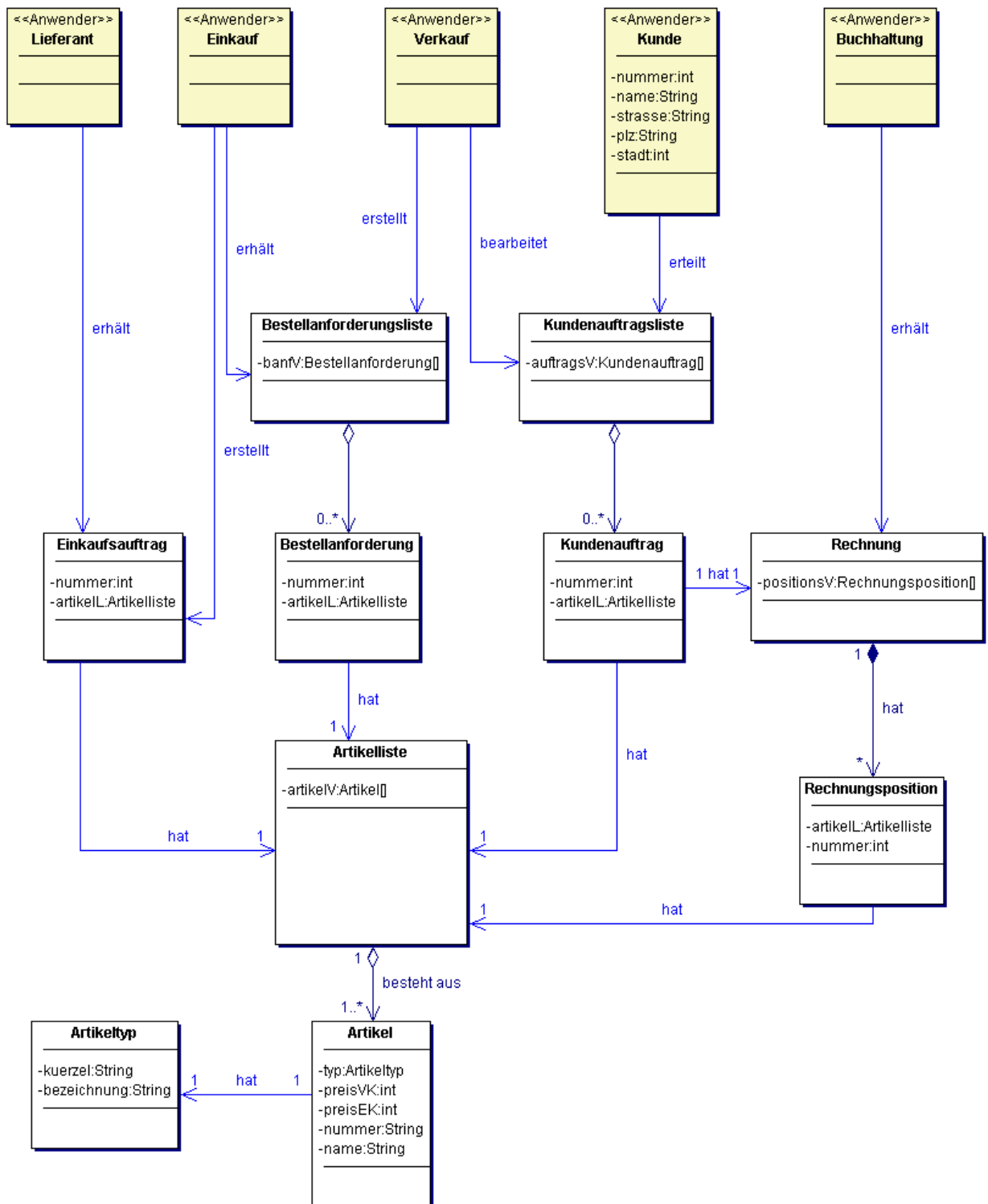


AF 60: Kundenauftrag ausführen



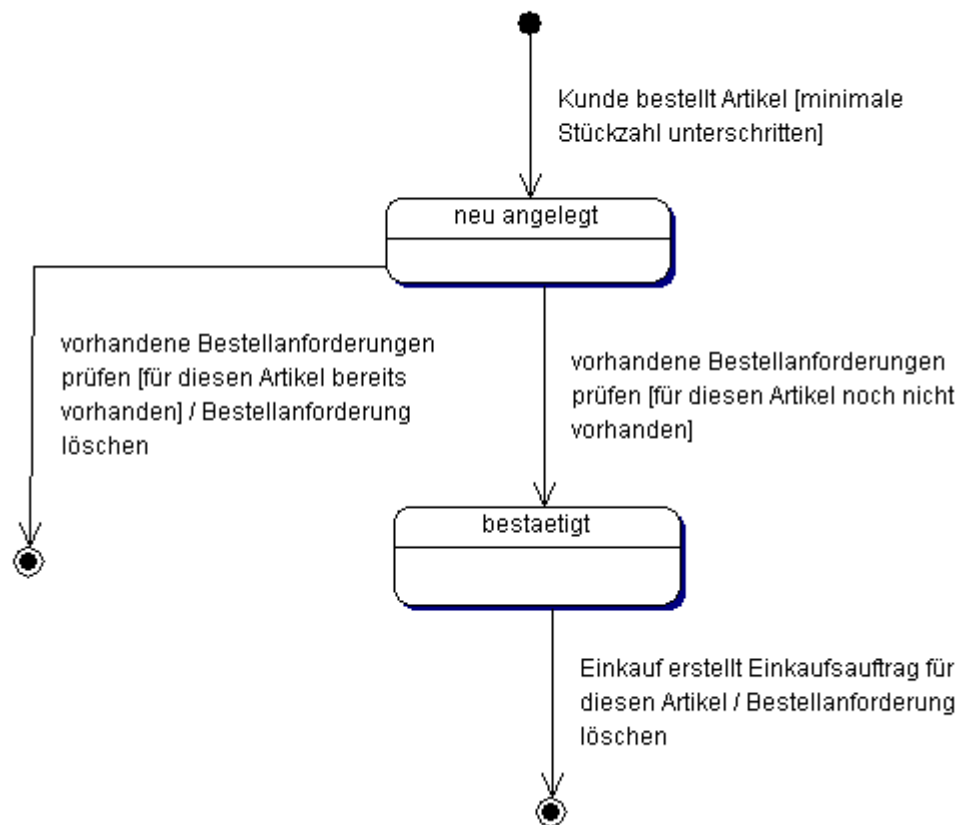
2 Analyse

2.1 Geschäftsklassendiagramm

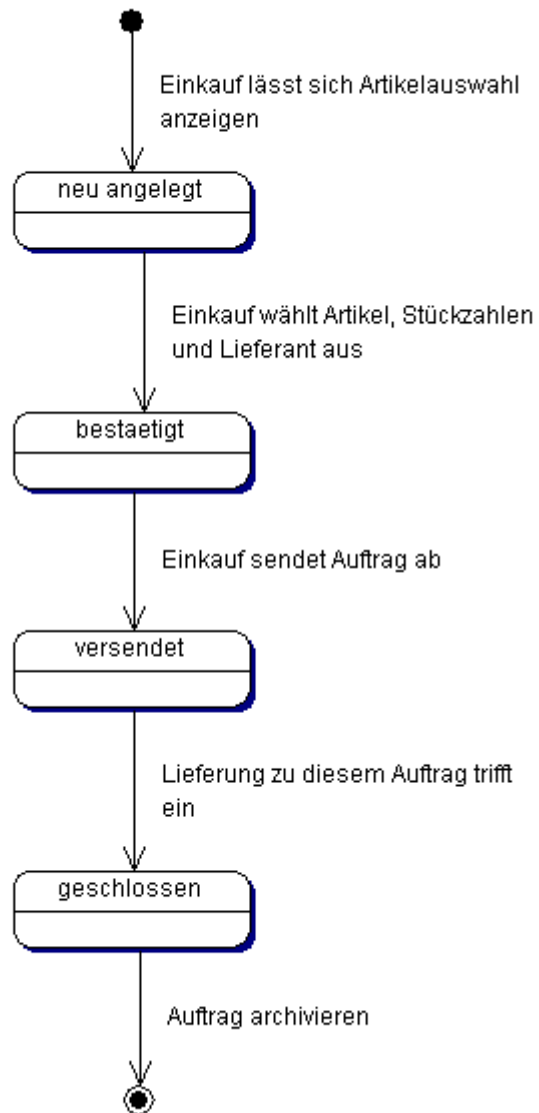


2.2 Zustandsdiagramme

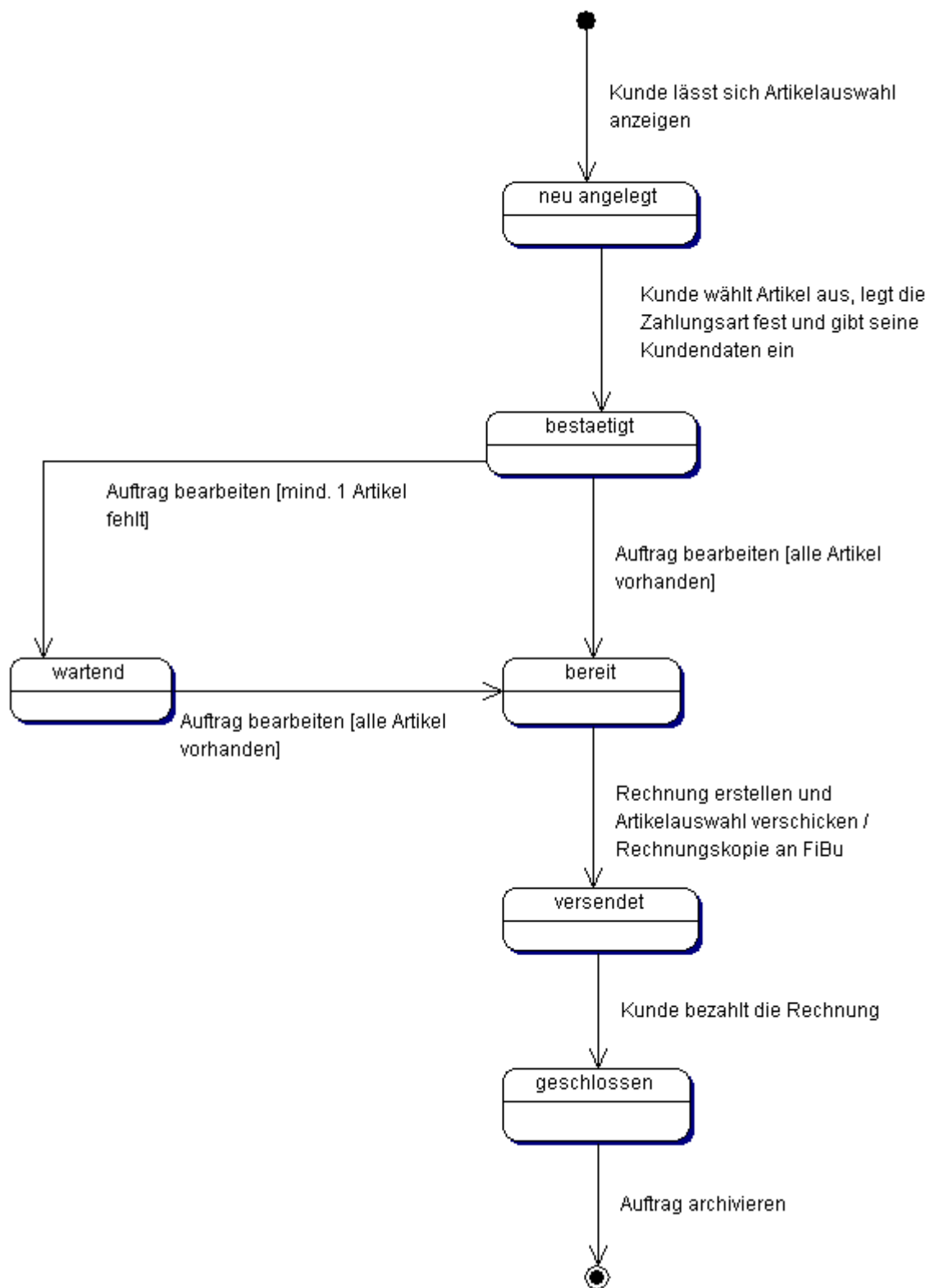
2.2.1 Zustandsdiagramm Bestellanforderung



2.2.2 Zustandsdiagramm Einkaufsauftrag



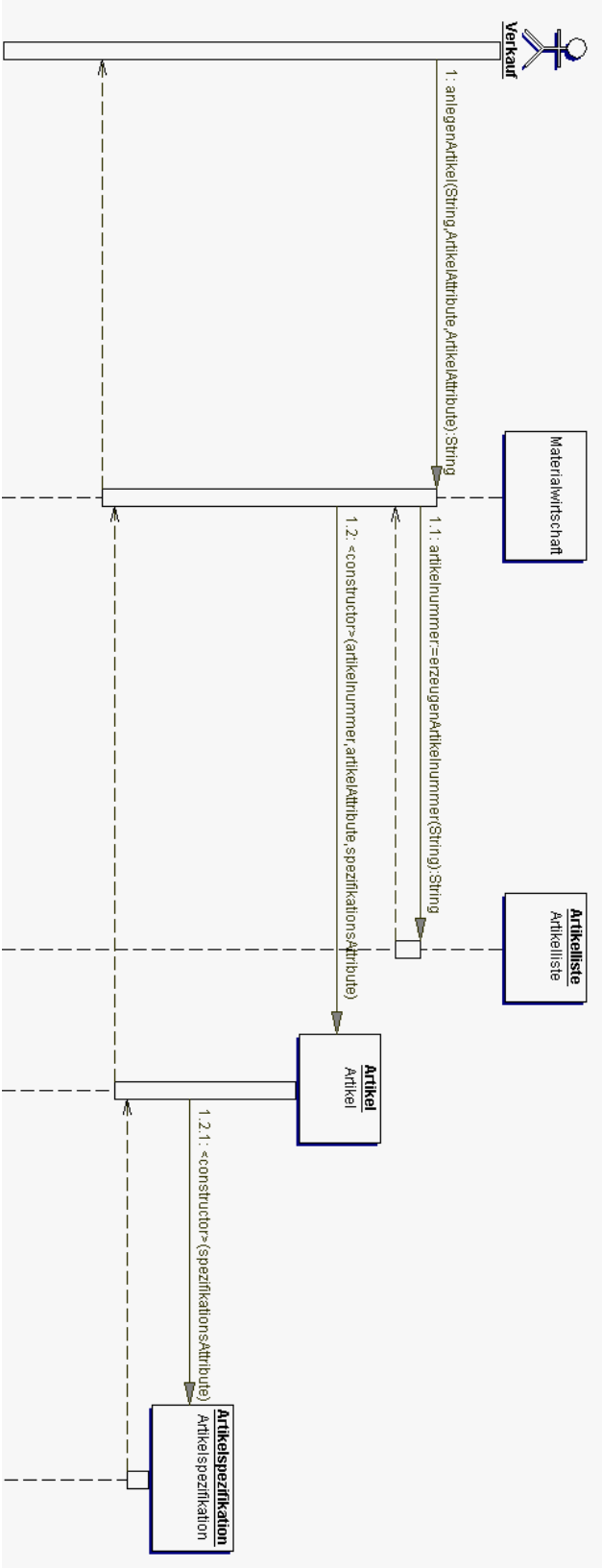
2.2.3 Zustandsdiagramm Kundenauftrag



3 Design

3.1 Sequenzdiagramme

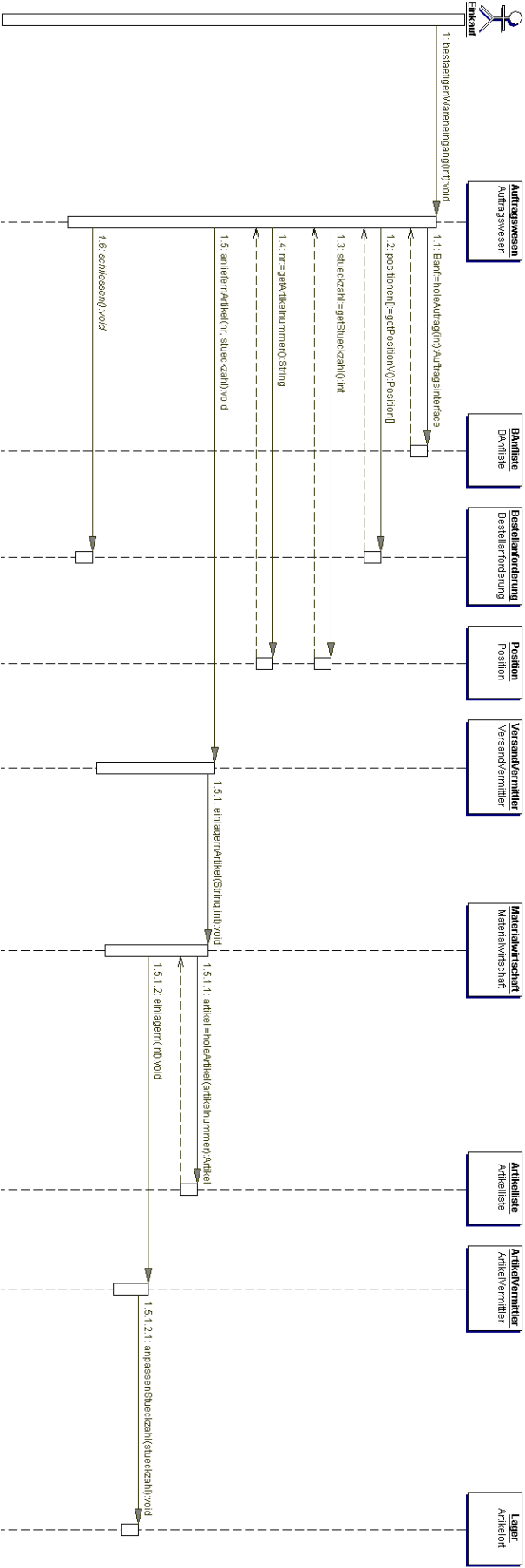
3.1.1 AF 10: Artikel anlegen



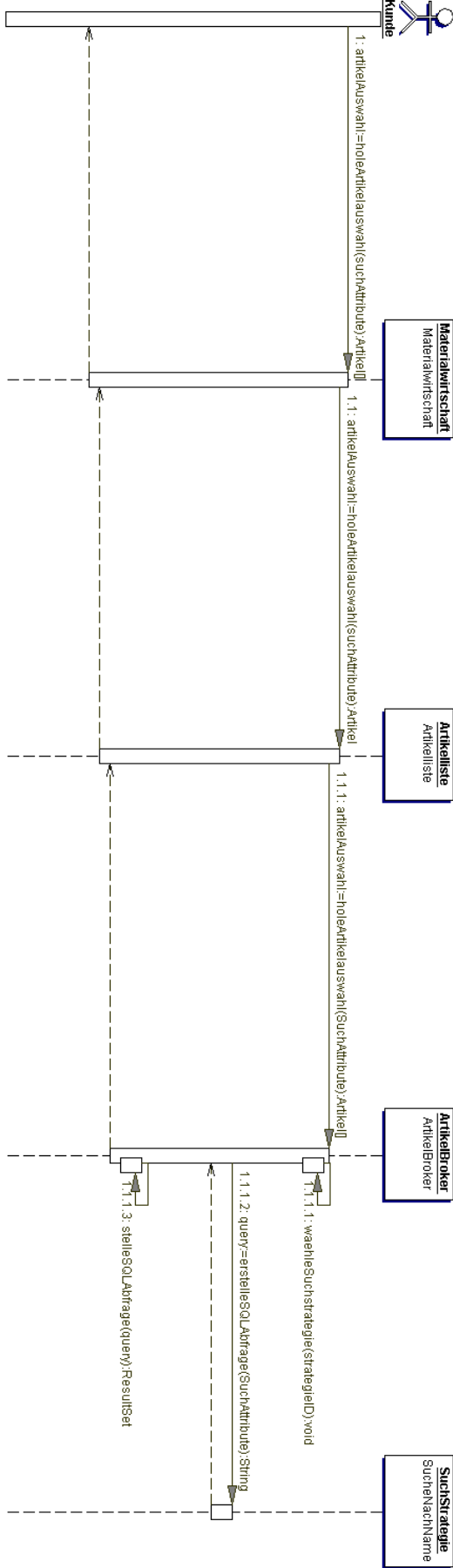
3.1.2 AF 20: Einkaufsauftrag anlegen



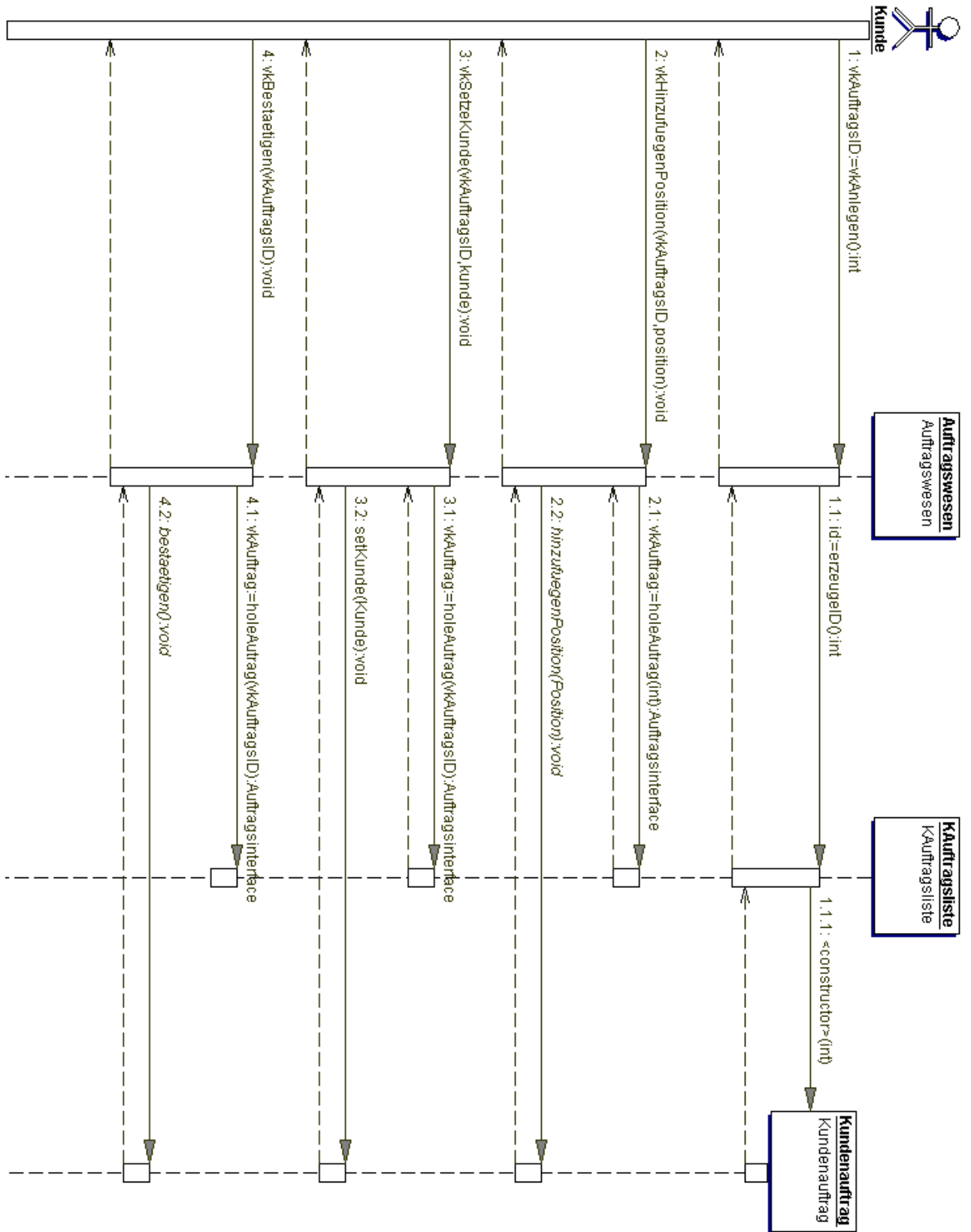
3.2.3 AF 30: Wareneingang bestätigen



3.1.4 AF 40: Artikelliste anzeigen

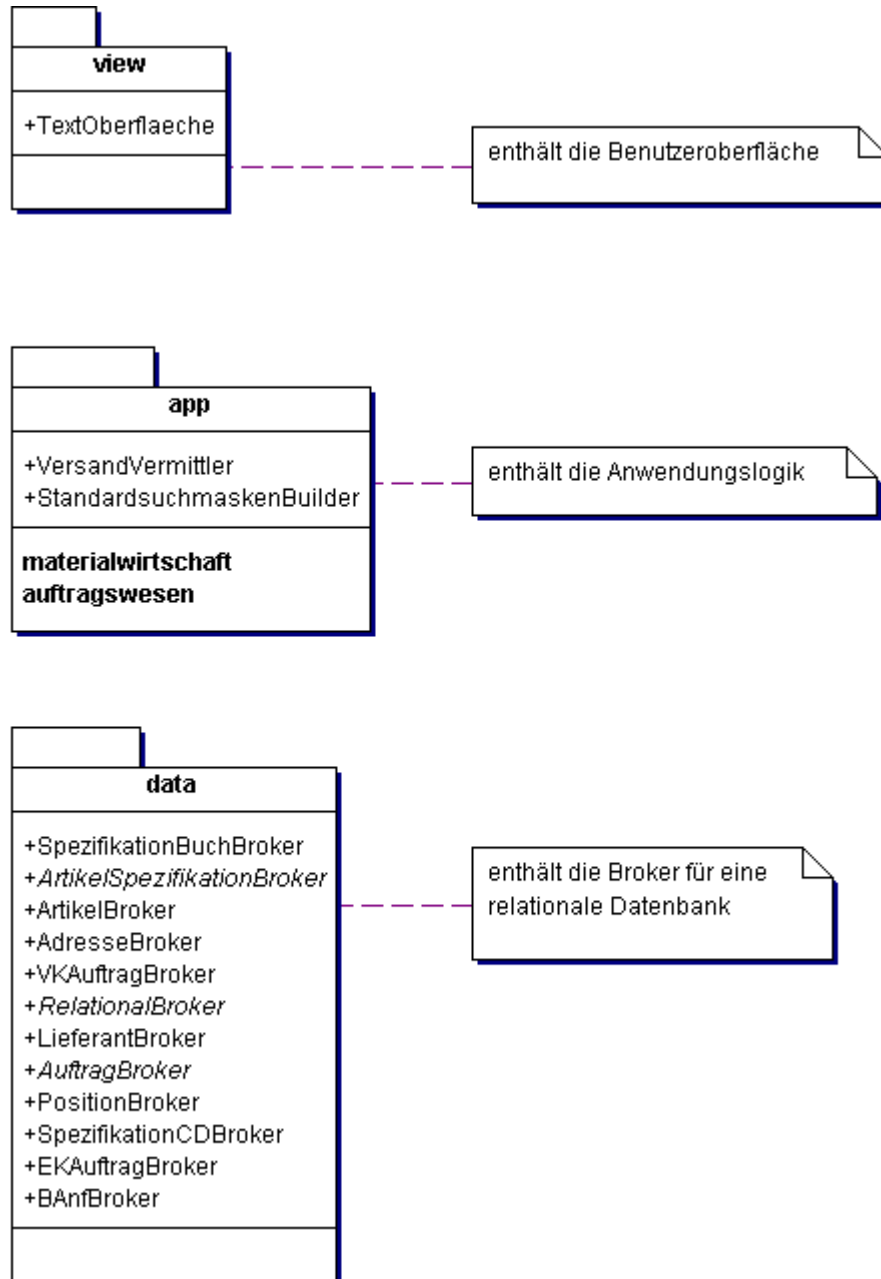


3.1.5 AF 50: Kundenauftrag erstellen

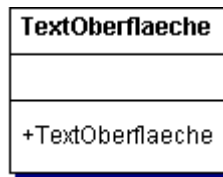


3.2 Systemklassendiagramme

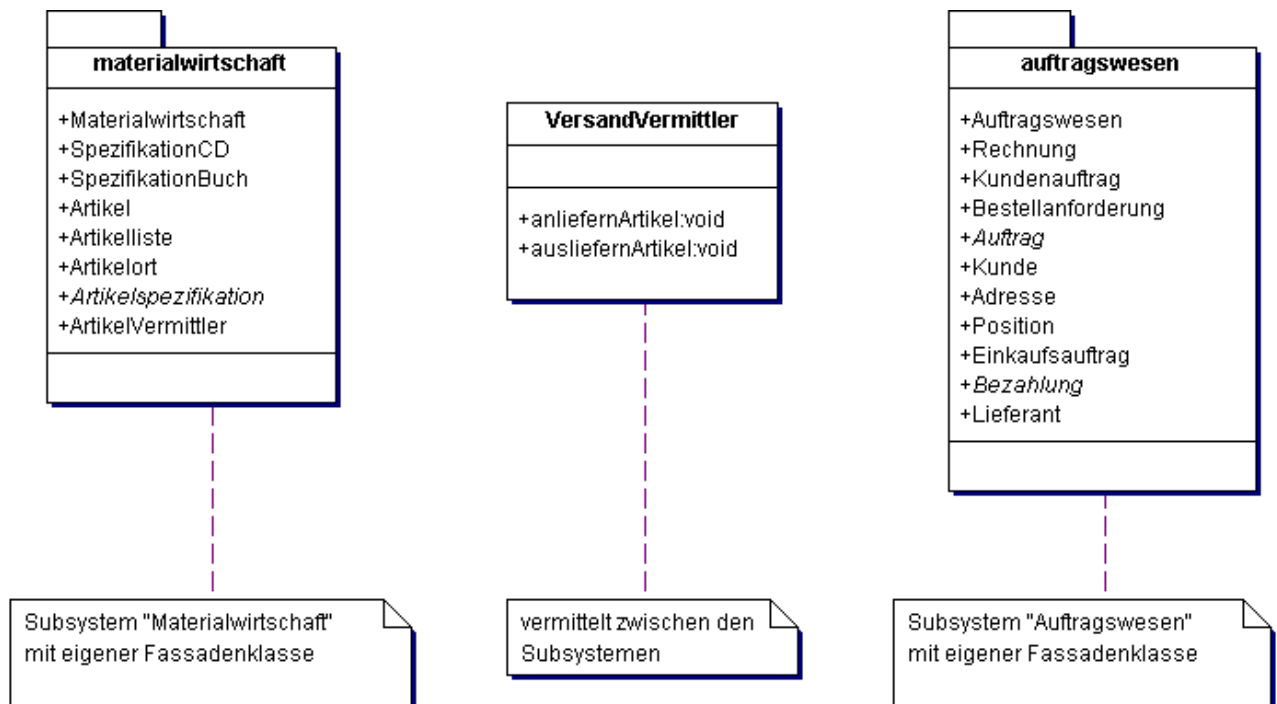
3.2.1 Package versand (Three-Tier-Architecture)



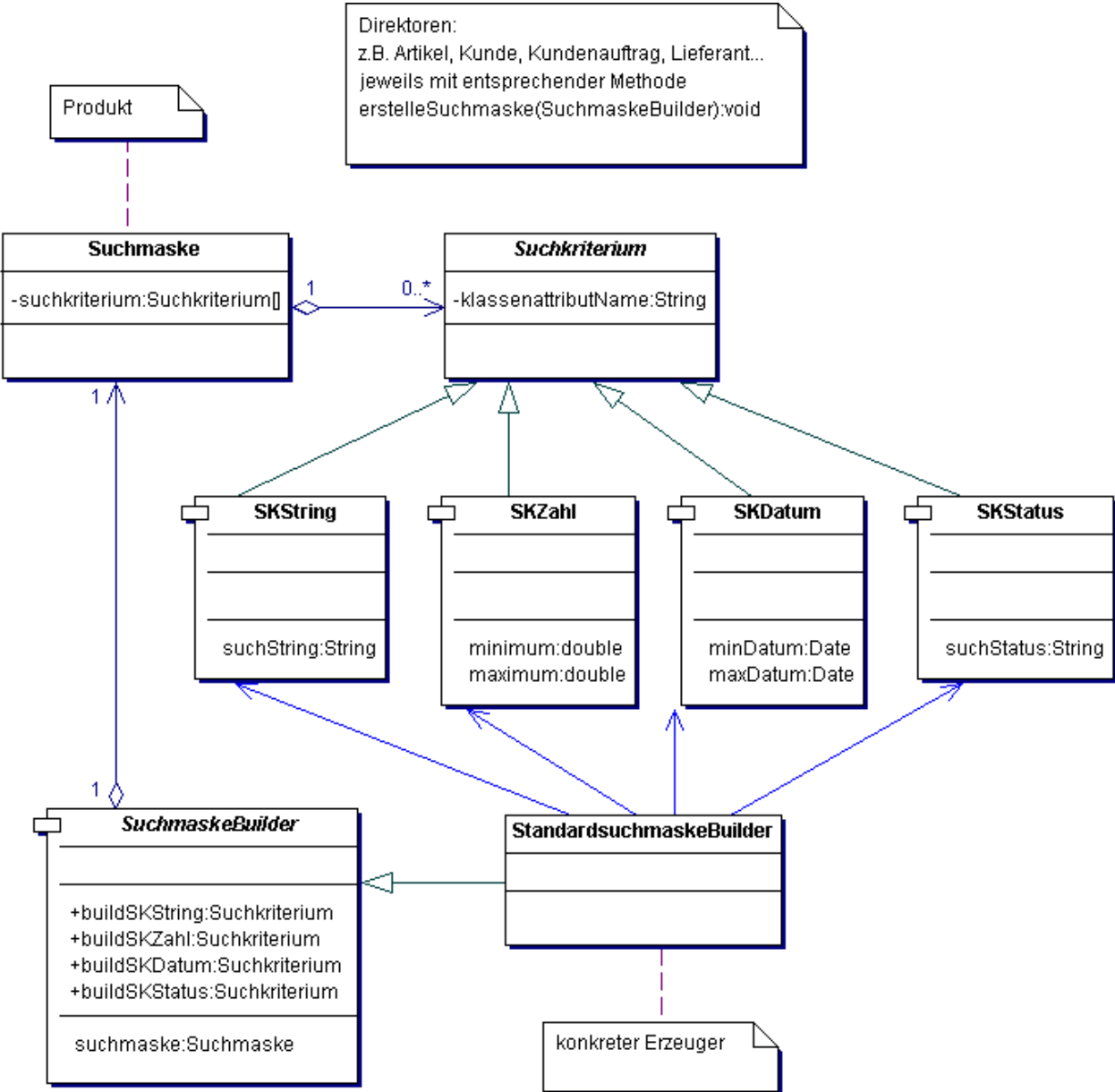
3.2.2 Package versand.view



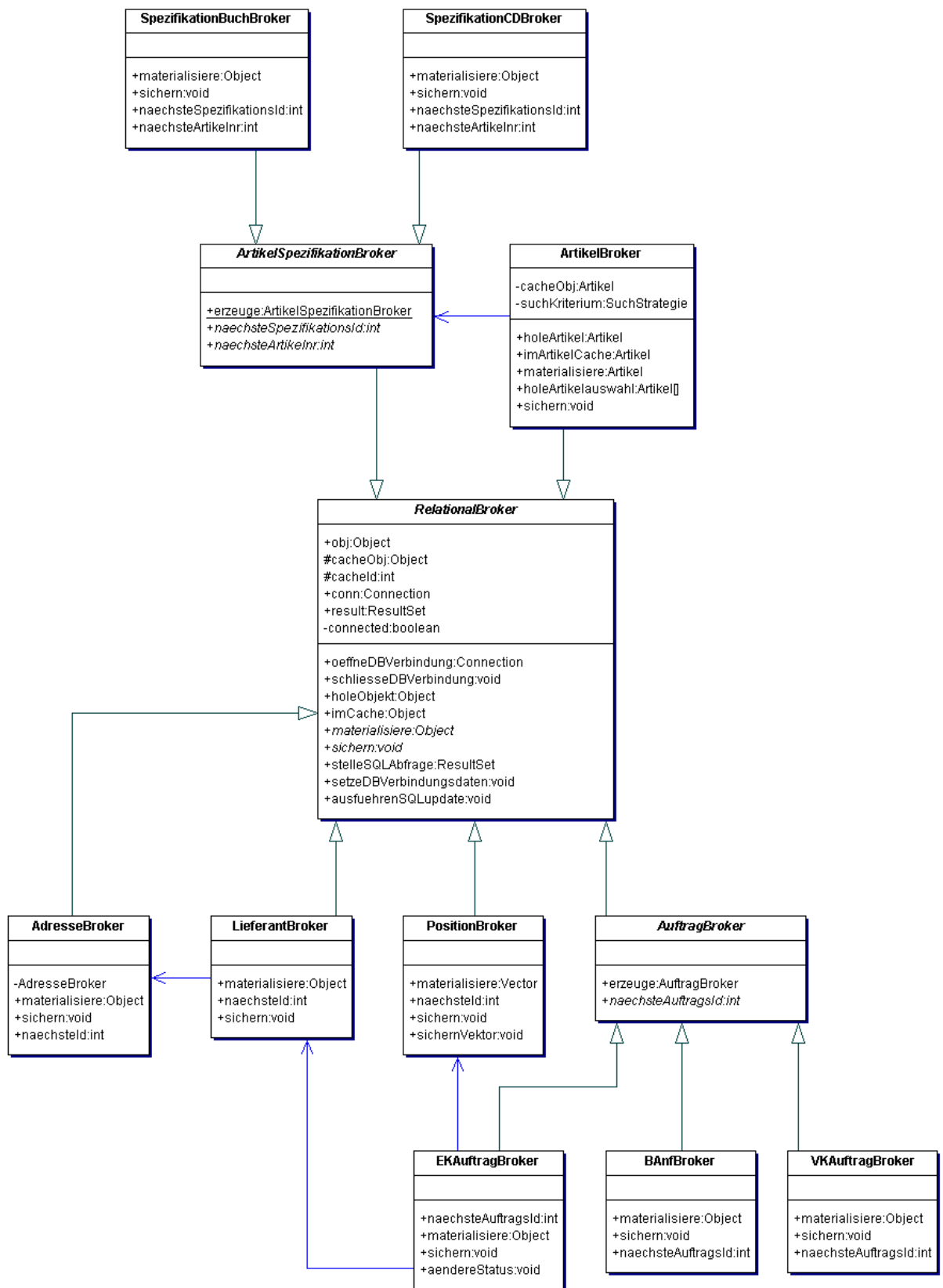
3.2.3 Package versand.app



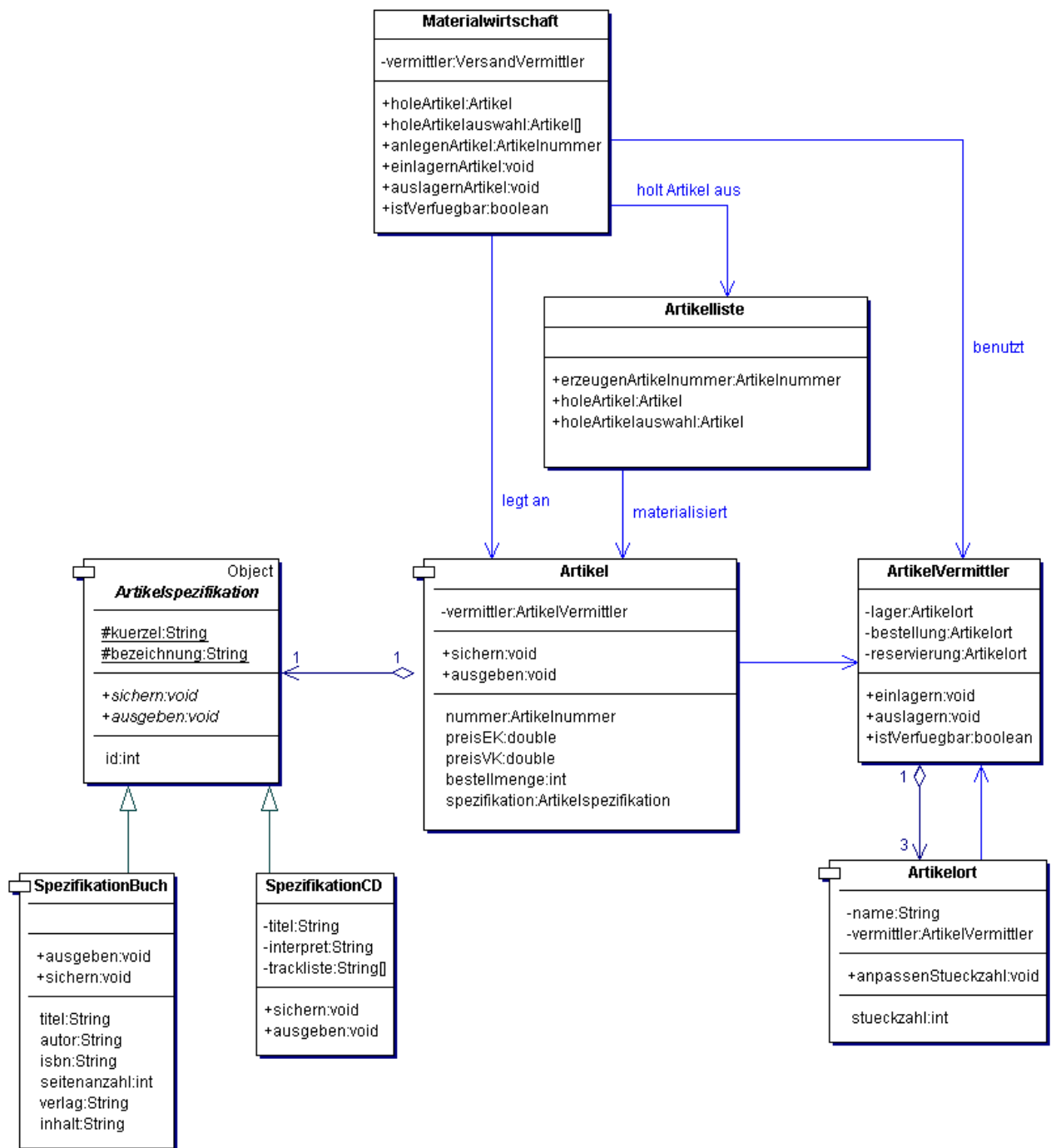
3.2.4 Package versand.app (Erzeugermuster für Erstellung von Suchmasken)



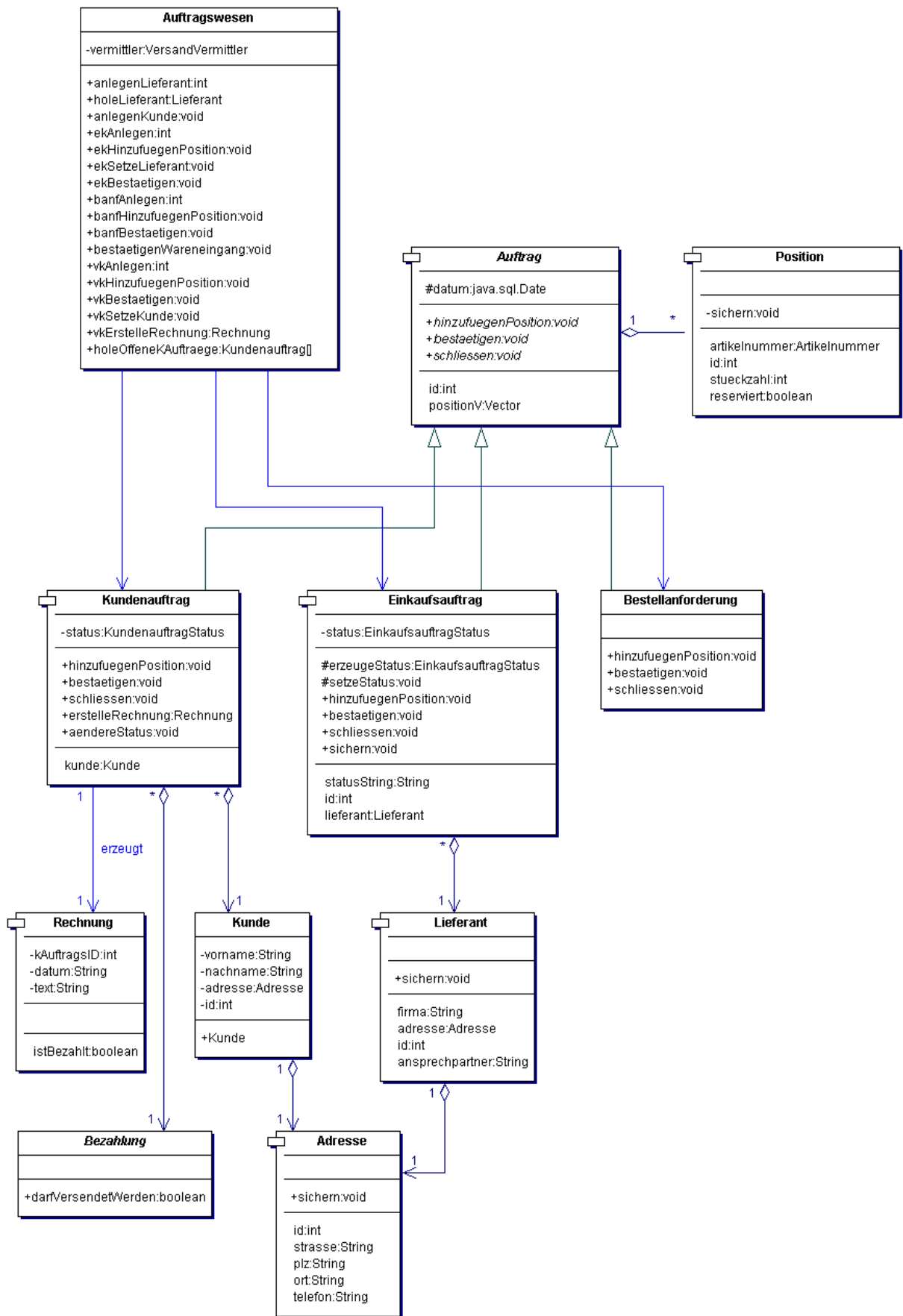
3.2.5 Package versand.data



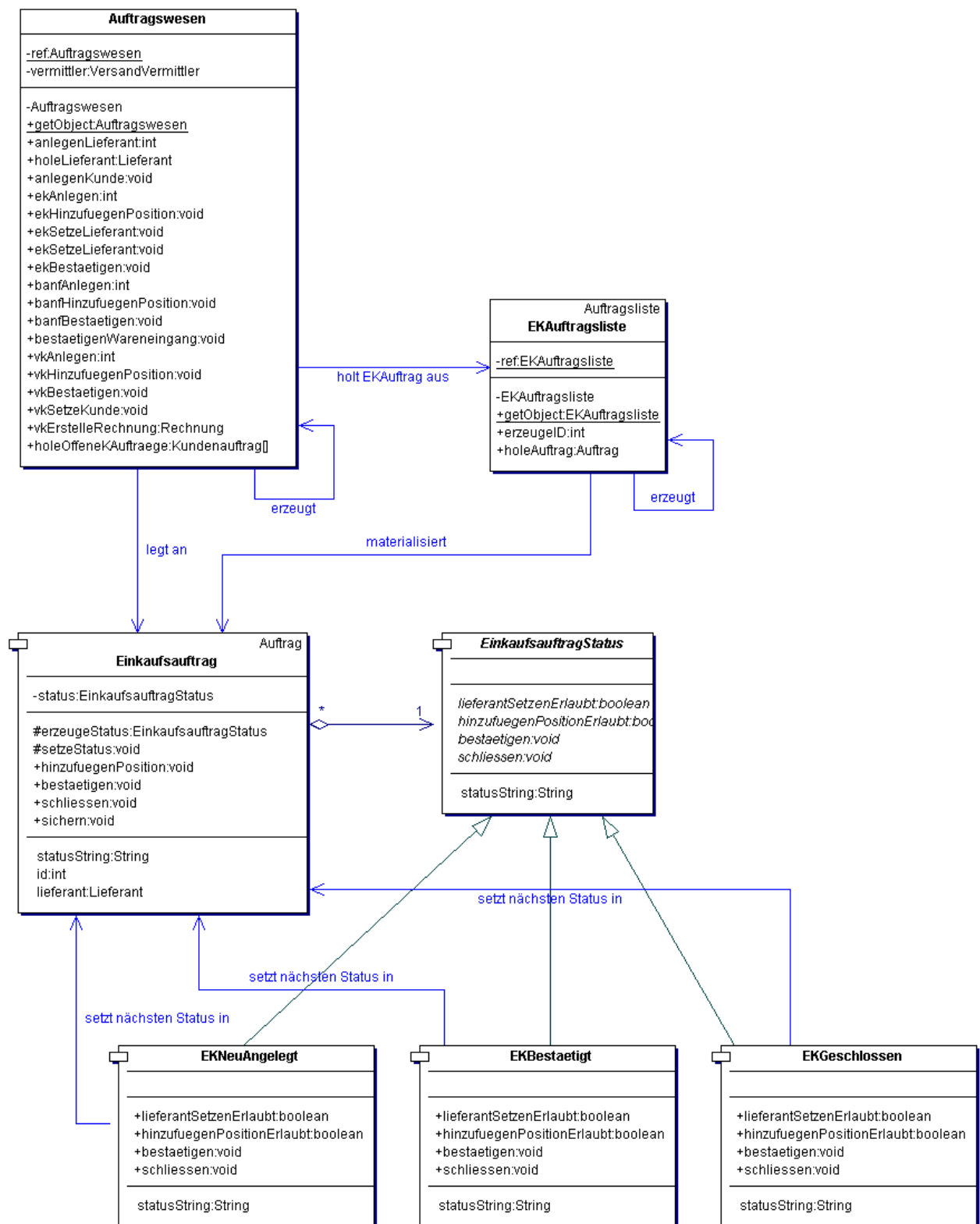
3.2.6 Package versand.app.materialwirtschaft



3.2.7 Package versand.app.auftragswesen



3.1.8 Package versand.app.materialwirtschaft (Zustandsmuster beim Einkaufsauftrag)



3.3 Erläuterungen zu verwendeten Entwurfsmustern

Verwendete Entwurfsmuster zur Erzeugung von Objekten: Erzeuger, Fabrikmethode, Singleton

Erzeuger

verwendet bei: Suchmaske

Das Erzeugermuster ermöglicht es, den genauen Aufbau eines Objekts zu verstecken und die Objektfabrikation an eine Direktor-Klasse zu delegieren. Hier werden als Direktoren diejenigen Klassen eingesetzt, für die jeweils eine Suchmaske erstellt werden können soll (z.B. Artikel, Kunde, ...). Diese Klassen wissen selbst am besten, für welche ihrer Attribute es sinnvoll ist, Suchkriterien zu erstellen.

Später könnten auch komplexere Filter implementiert werden, weshalb die konkrete Instanzierung von Suchmasken an Erzeugerklassen delegiert wird. Dies ermöglicht eine große Flexibilität für Erweiterungen in der Filterfunktion für die Zukunft.

Allerdings müssen die Broker, die letztendlich für die Umsetzung der Suchkriterien sorgen müssen, die Suchattribute sehr genau inspizieren und mit großem Aufwand die Suchkriterien in ihre Funktionalität integrieren. Es besteht dann ein hoher Zusammenhalt zwischen Broker und Suchmaske, was aber wohl systembedingt ist und nicht so einfach geändert werden kann.

Fabrikmethode

verwendet bei: Artikelspezifikation(Broker), AuftragBroker

Die Fabrikmethode der (abstrakten Klasse) Artikelspezifikation ermöglicht es Klienten, ohne Wissen über die geforderte Subklasse eine Instanz einer konkreten Artikelspezifikation zu erhalten. Das zugehörige Kürzel des Artikeltyps (bestehend aus drei Buchstaben) reicht dafür aus. Beim ArtikelspezifikationBroker wird die Fabrikmethode analog verwendet.

Singleton (Einzelmuster)

verwendet bei: Materialwirtschaft, Auftragwesen, allen Vermittlern, allen Brokern

Wird überall dort verwendet, wo nur eine einzige Instanz der Klasse während der gesamten Laufzeit benötigt wird. Die hier verwendete einfache Implementierung des Singleton-Musters ist nicht thread safe, was aber ausreicht, da das gesamte System keine Threads verwendet.

Verwendete Entwurfsmuster für das Verhalten von Objekten: Zustand, Vermittler

Zustand

verwendet bei: Bestellanforderung, Einkaufsauftrag, Kundenauftrag

Wird verwendet, wenn Objekte stark unterschiedliches Verhalten je nach ihrem inneren Zustand zeigen, oder wenn es viele verschiedene Zustände gibt. Die Zustandsübergänge sind an wohldefinierten Stellen und können an die Zustandsobjekte delegiert werden.

Bei den verschiedenen Auftragsarten haben wir das Zustandsmuster verwendet, da sie sowohl stark unterschiedliches Verhalten je nach Zustand zeigen, als auch viele verschiedene Zustände haben (Kundenauftrag: sechs Zustände).

Problematisch ist hier höchstens, dass die Aufträge kaum mächtige Methoden haben, sondern hauptsächlich get- und set-Methoden für die verschiedenen Attribute (z.B. Positionen, Lieferant) bereitstellen. Die Umsetzung solcher Methoden ist im Statusmuster schlecht möglich, da der Kontext eine set-Anforderung an sein Statusobjekt weitergibt, was dann wiederum eine „entsprechende“ Methode im Kontext aufruft - also wiederum dieselbe Methode, die das Statusobjekt aufgerufen hatte. Für diese Methoden haben wir uns damit beholfen, entsprechende „setAttributIstErlaubt:boolean“ Methoden in die Statusklassen zu implementieren.

Vermittler

verwendet bei: VersandVermittler, ArtikelVermittler

Das Vermittlermuster ermöglicht es verschiedenen Subsystemen, miteinander zu kommunizieren, ohne direkt auf das jeweils andere System zuzugreifen. Beim Versandhandel müssen die Subsysteme Materialwirtschaft und Auftragswesen bei den Anwendungsfällen „Wareneingang bestätigen“ und „Kundenauftrag ausführen“ miteinander kommunizieren. Um eine Entkopplung der Systeme zu ermöglichen, wird hier der „VersandVermittler“ eingesetzt. Wir haben uns dafür entschieden, den Vermittler sehr einfach zu halten und nur die nötigsten Funktionalitäten dort zu implementieren. Das Wissen um die Abwicklung zum Beispiel eines Wareneingangs befindet sich deshalb größtenteils in den Fassadenklassen der Subsysteme.

Der ArtikelVermittler wird eingesetzt, um eine Entkopplung zwischen der Klasse Artikel und den verschiedenen Artikelorten zu ermöglichen. Die Funktionalität der (Um)verteilung von Artikelstücken wird in den Vermittler verlagert. Weder muss der Artikel etwas über seine Orte wissen, noch müssen die Orte etwas über die Artikel wissen. Auch ist die Materialwirtschaft somit relativ leicht um weitere Artikelorte erweiterbar.

Verwendetes Entwurfsmuster für die Struktur von Klassen: Fassade

Fassade

verwendet bei: Auftragswesen, Materialwirtschaft

Für die Subsysteme Auftragswesen und Materialwirtschaft wurden Fassadenklassen eingerichtet, um Klienten die (besonders beim Auftragswesen) auf viele Klassen verteilte Funktionalität in einer einzigen Klasse zur Verfügung zu stellen.

4 Implementierung

Pfad zur lauffähigen Implementation:

`/home5/g89/std8945/public_html/ooad/`