Softwaretechnik
2002
Prof. Dr. H. Klaeren und M E Leypold

Eberhard Karls
Universität
Tübingen


Übungsblatt 6

Statische OO-Modelle

Ausgabe: 11.6.2002
Abgabe: 20.6.2002, 12:00

Inhaltsverzeichnis

1 Übersicht

Thema: Gegenstand dieses Übungsblatts ist die statische *objektorientierte Modellierung* – mit einem Wort: Klassendiagramme. Ich werde zuerst einige, hoffentlich hilfreiche, allgemeine Hinweise zur Erstellung von (guten) Klassendiagrammen geben. Aufgabe 1 besteht im Wesentlichen aus einzelnen Fingerübungen zur Objektmodellierung mit Klassendiagrammen, während in Aufgabe 2 ein Datenmodell für einen Ausschnitt eines Web-Shops erstellt werden soll.

Bewertung: Aufgabe 1 werden wir wieder mit etwa einem Viertel bis einem Drittel der auf dieses Blatt erhältlichen Punkte werten. Wie immer ist diese Angabe vollkommen unverbindlich, da wir uns vorbehalten möchten, die Bewertung der Situation anzupassen.

Hinweis zum Bearbeitungszustand: *Die Hinweise aus dem nun folgenden Kapitel sind umfangreich genug, um als eigenständiges Handout ausgegliedert zu werden, wurde hier jedoch noch im Aufgabenblatt belassen, um in (weitgehender) Übereinstimmung mit den ursprünglich während des Semesters ausgegebenen Aufgabenblättern zu bleiben. Die Kapitelnummerierung wurde geändert (Die Aufgaben heißen nun auch A und B), der Wortlaut der Aufgaben weitgehend beibehalten.*

2 Hinweise

2.1 Klassendiagramme

Klassendiagramme beschreiben Daten und Beziehungen zwischen Daten, sowie zugeordnete Operationen. Letzteres soll hier allerdings noch im Hintergrund bleiben.

Daten sind in Objekten gespeichert. Beziehungen stehen zuerst einmal für entscheidbare Aussagen über Objekte, wird z. B. zwischen den Klassen *A* und *B* eine Assoziation (Beziehung) mit dem Namen „frozzles“ definiert, so muss zu einem Zeitpunkt t_1 für ein Objekt *a* aus der Klasse *A* im Zustand α_1 entscheidbar sein, ob es zum Objekt *b* aus der Klasse *B*, welches jetzt den Zustand β_1 hat, in der Beziehung „frozzles“ steht – mit einem Wort, ob die Aussage *a frozzles b* wahr ist.

Während der Modellierung liegen viele Attribute und Assoziationen nur „im Auge des Betrachters“, d. h. über deren spätere Entsprechungen in der Implementation wird noch nichts impliziert. Insbesondere darf eine Assoziation bzw. eine Aggregation nicht einfach mit einem Zeiger oder einem Array identifiziert werden.

Es sei zudem erwähnt, dass damit auch noch nicht gesagt ist, ob ein Objekt die Möglichkeit besitzt, Assoziation abzufragen (zu *navigieren* wie das im UML-Jargon heißt), also z. B. ein Mechanismus existiert, mit dem *a* feststellen kann, zu welchen Objekten aus *B* es in der Beziehung „frozzles“ steht.

2.2 Einschränkungen (Constraints)

Ein Datenmodell charakterisiert alle gültigen Zustände für ein Programm oder Teilaspekte eines größeren datenverarbeitenden Systems: Es trifft klare Aussagen, welche Zustände möglich und zulässig sind, und vor allem auch, welche Zustände, obwohl von der der Mechanik der Programmiersprache her möglich, dennoch im Betrieb des in Frage stehenden Softwaresystems nie auftreten *dürfen*.

Da Klassendiagramme ein vollständiges Datenmodell ermöglichen sollen, folgen auch sie dem üblichen Muster von Konstruktion und Selektion (Constraint), das wir schon aus der Mathematik

$$M = \{x \in A \times B \mid x \text{ is in bounds}\}$$

und aus VDM-SL kennen:

```
types
  M ::
    a : A
    b : B
  inv m == IsInBounds(m.a,m.b);
```

In diesen Beispielen übernimmt jeweils das kartesische Produkt „×“ bzw. die Bildung eines Composite-Typs die Rolle der Konstruktion einer neuen (Zustands-) Menge, aus welcher dann durch das Prädikat hinter dem „|“ bzw. durch die Invariante eine interessante Untermenge ausgewählt wird.

In analoger Weise können und dürfen, wenn das Klassendiagramm ein einigermaßen trennscharfes Modell entwerfen soll, die Objekte einer Klasse nicht beliebige Attributwerte annehmen (solange das System intakt ist) und auch Assoziationen unterliegen gewissen Bedingungen, d. h. sie treten nur unter bestimmten Umständen auf.

Solche Einschränkungen könnten in OCL (der *Objekt Constraint Language* notiert werden, üblicher ist es zur Zeit jedoch noch, natürlichsprachige Erläuterungen zu Assoziationen und Attributen hinzuzufügen. Dies kann in Form eines Diagrammelements *Notiz* geschehen, oder das Hinzufügen einer kurzen Bemerkung in Schweifklammern („{ … }“) an die entsprechenden Stellen. Für längere Bemerkungen schlage ich vor, im Stil von Fußnoten eine Verweismarke in Schweifklammern anzubringen und die Einschränkungen und Bedingungen in einer vom Diagramm getrennten Liste zu notieren.

2.3 Dynamische Aspekte

Gewisse dynamische Aspekte sind nicht selten auch in Klassendiagrammen von Interesse, können dort jedoch nicht graphisch notiert werden, lohnen aber in einfachen Modellen die Erstellung eines Sequenz-Diagramms nicht. Beispiel: Objekte der Klasse *B* erzeugen unter bestimmten Umständen Objekte der Klasse *A*. „Erzeugt werden“ oder „erzeugt worden sein“ ist aber keine sinnvoll feststellbare Relation zwischen Objekten (man beachte, dass wir dazu nur immer die *Situation* zu einem bestimmten Zeitpunkt betrachten dürfen): Hier wird ein Ablauf gekennzeichnet, keine dauernde Beziehung.

Ein anderes Beispiel wäre eine Zusicherung, dass gewisse Attribute eines Objektes über sein Lebensdauer hinweg konstant bleiben (z. B. die in einen Personalausweis eingedruckten Personenmerkmale). Auch dies ist eine dynamische Eigenschaft.

Auch solche dynamischen Aspekte, wie die eben erläuterten, können durch natürlichsprachige Annotationen in einem Klassendiagramm verankert werden.

2.4 Operationen

Operationen gehören im Grunde zu den dynamischen Aspekten des Systems. Da sie jedoch einzelnen Klassen zugeordnet sind, wird ihre Existenz in Klassendiagrammen festgehalten.

Tendenziell sollten Operationen erst nach der statischen Modellierung gesucht und charakterisiert werden. Einige Operationen drängen sich aber bereits während der statischen

Modellierung geradezu auf, und werden (können) bereits dann festgehalten und charakterisiert werden. Da die meisten OO-Prozesse iterativ sind, ist es sowieso schwierig, statische und dynamische Entwurfstätigkeiten zeitlich voneinander zu trennen. Trotzdem bleibt auch hier die Faustregel „Die Daten zuerst!“ gültig.

Häufig lässt sich die Leistung einzelner Operationen einfach in natürlicher Sprache charakterisieren, indem man Vor- und Nachbedingungen und veränderte Zustände beschreibt (hier ist er wiederum: Der Geist der formalen Spezifikation), beispielsweise:

- Setzt Attribut a auf $a > x$ und $a < y$ und lässt alle anderen Attribute konstant.
- Hebt die Assoziation „frozzles“ mit allen Objekten de Klasse *B* auf.
- Darf nur im Zustand $count > 0$ gerufen werden und erzeugt dann ein Objekt der Klasse *B*.

Solche natürlichsprachigen oder informell mathematischen Beschreibungen können wie Einschränkungen oder andere dynamische Aspekte als Notizen oder Fußnoten notiert werden.

2.5 Vererbung

Vererbung ist - obwohl für den Anfänger das herausstechendste Merkmal der OO-Techniken – am Anfang der Modellierung von geringem Wert und sollte da nicht eingesetzt werden.

Vererbung wird im Wesentlichen mit zwei Zielen eingesetzt:

1. Zur Formalisierung einer *Can-be-used-as*-Beziehung.
2. Zur Wiederverwertung von Implementationsteilen.

Erstes bezieht sich deutlich auf die Rollen, die Objekte spielen, bzw. in der sie von Algorithmen behandelt werden. Diese Art der Generalisierung kann erst nach einer verhältnismäßig weit fortgeschrittenen *dynamischen* Modellierung (wir reden hier ja von Operationen!) vernünftig durchgeführt werden. Zweiteres ist bereits eine Implementationstechnik und soll auf keinen Fall in den Entwurf zurückwirken.

Vererbungsbeziehungen erzeugen sehr starke Abhängigkeiten zwischen Systemteilen und müssen deshalb so spät als möglich eingeführt werden.

In diesem Übungsblatt gibt es keinen sinnvollen Einsatz für eine Vererbungsbeziehung!

2.6 Zur Verb-Substantiv-Analyse

Kunden und Informatiker tendieren im allgemeinen dazu, lieber Vorgänge (Prozesse) als Situationen zu beschreiben. Die Kunst der statischen Modellierung besteht genau darin, aus Beschreibungen von Abläufen einen Systemzustandsbegriff, ein Datenmodell, herauszuschälen. Die Verb-Substantiv-Methode bietet hier einen geeigneten Ansatzpunkt (Substantive entsprechen heuristisch Daten, die Verben den Operationen, den Prozessen). Allerdings kann diese Methode auch in die Irre führen: Nicht jedes Substantiv eignet sich als Klasse und nicht jedes Substantiv ist für den zu modellierenden Ausschnitt des Systems wirklich relevant. Hier ist sorgfältige Abwägung gefragt.

Außerdem sollte man der Suggestion der Klassennamen nicht erliegen und „ontologisch“ modellieren: Wir reden im Kontext des Entwurfs von Software in der Regel nur über Konstrukte innerhalb des Computers, d. h. *nicht* über die reale Welt. Ein Objekt *a* der Klasse

Auto ist nicht mein Auto, sondern es *beschreibt* (in diesem Fall) mein Auto. Und „Flugzeugträger“ und „Fischkutter“ mögen draußen in der Welt beides „Schiffe“ sein – was keineswegs beduetet, dass in der Software hier eine Vererbungsbeziehung enthalten sein *muss*.

Objekte, die die Namen physikalischer Gegenstände tragen, sind im Kern meist nur Datensätze, die diese Gegenstände beschreiben[1]. Modelliert werden nur die Beziehungen zwischen Datensätzen.

Es lohnt, sich diesen Sachverhalt zuweilen vor Augen zu führen, gerade, weil es soviel schlechte OO-Literatur gibt, die diesen Unterschied nicht macht.

3 Aufgabe A: Aufwärmübungen

In den folgenden Fingerübungen zur Objektmodellierung sollt Ihr immer kleine, prototypische Fälle einzelner Konstrukte aus Klassendiagrammen durchspielen.

Erstelle bitte für jeden der folgenden Fälle ein eigenes Diagramm, auch wenn sich viele Beispiele um das gleiche oder ein ähnliches Thema drehen.

1. Der Zustand eines Objektes einer Klasse wird durch seine Attribute beschrieben. Modelliere: Ein Schweizer Nummernkonto wird durch Kontonummer, Bankleitzahl und Kontostand beschrieben.

2. Attribute unterliegen häufig gewissen Einschränkungen bzw. gegenseitigen Abhängigkeiten. Das bedeutet: Attribute können nicht jeden beliebigen Wert annehmen. Modelliere: Einem Konto ist ein Überziehungskredit zugeordnet. Der Kunde darf den Überziehungskredit nicht unterschreiten.

3. Modelliere: Aus der Sicht der Schweizer Bank gibt es zu jedem Nummernkonto einen oder mehrere Verfügungsberechtigte, die nur durch ihr persönliches Kennwort identifizierbar (bzw. bekannt) sind (Vorsicht Falle! Denke genau über die Kardinalitäten nach).

4. Modelliere: Aus der Sicht des Finanzamtes unterhält jeder der Steuerhinterziehung Verdächtige ein oder mehrere Konten. Ein Verdächtiger wird durch Name, Geburtsdatum und Steuernummer identifiziert.

   Aus diesem und dem letzten Beipiel kann man lernen, dass es nicht um die Modellierung der Realität (was immer das ist :-) geht, sondern darum, den Datenbestand aus der Sicht des jeweiligen Problems zu beschreiben, das heißt, relevante Ausschnitte und Perspektiven zu bilden.

5. Attribute unterliegen unter Umständen gewissen Einschränkungen, die aus dem Zustand / Kontext herrühren, in den ein Objekt eingebettet ist. Modelliere: Jedem Konto ist ein Buchungsgeschichte zugeordnet. Die Buchungsgeschichte besteht aus einzelnen Buchungen. Buchungsgeschichte und Kontostand müssen konsistent sein.

6. Manche Assoziationen sind von anderen Assoziationen abhängig – d.h. die Assoziation liegt nur vor (oder eben gerade nicht vor) wenn bestimmte andere Assoziationen vorliegen. Modelliere: Wenn A auf das Konto einzahlt und B vom Konto abhebt, kann man davon ausgehen, dass A zu B geschäftliche Beziehungen unterhält.

Abzugeben sind: 6 Klassendiagramme (zu den obigen Situationen) mit den nötigen Erläuterungen (Constraints, dynamische Aspekte), falls nötig und zutreffend.

[1] Ausnahmen sind Proxy-Objekte, die den Zustand physikalischer Teilsysteme (wie externe Sensoren) kapseln.

4 Aufgabe B: Statisches Modell eines Web-Shops

Modelliere den Datenbestand des im Folgenden beschriebenen Web-Shops mit allen Klassen, Attributen, Assoziationen und Einschränkungen auf den Attributen und Assoziationen.

Die Firma *United Clothes* produziert Kleidung und Textilartikel und vertreibt nur an Wiederverkäufer. Zu den Kunden von *United Clothes* gehören Warenhäuser, Supermärkte und der Einzelhandel. Seit einigen Jahren sind – bedingt durch einen verschärften internationalen Konkurrenzkampf – die Umsätze im Schwinden begriffen. *United Clothes* hofft nun, die Händler durch ein neues, vorteilhaftes Vertriebskonzept an sich binden zu können. Bestandteil dieses Konzepts ist ein B2B-Web-Shop[2], in dem die Händler einen Katalog online durchsuchen und direkt bestellen können.

Die folgenden Abschnitte beschreiben verschiedene Aspekte des Ablaufs vom Login bis zur Rechnungsstellung.

4.1 Login, Sessions und Warenkörbe

Im Folgende wird die bestellende Person oder Abteilung als *Kunde* bezeichnet. Wir werden später noch sehen, dass dies nicht notwendig die Person ist, die die Rechnung bezahlen wird.

Wenn der Kunde den Web-Shop betritt, wird ihm ein Login-Formular präsentiert, auf das er Login-Namen und Passwort einträgt. Dieses wird mit den Eintragungen in der Kundendatenbank verglichen. Hat sich der Kunde korrekt authentisiert, wird eine Session erzeugt. Alle weiteren Operationen innerhalb des Shops (Abrufen von Information und insbesondere Veränderungen des Warenkorbes) finden im Kontext dieser Session statt. Das bedeutet konkret: Auf dem Rechner des Kunden wird ein eindeutiges Cookie gespeichert, das bei jedem Aufruf einer Seite wieder mit zurück an den Server übertragen wird. Das Cookie dient dazu, die Session zu identifizieren, welche wiederum auf den Kunden verweist. Jeder Kunde besitzt genau einen Warenkorb auf dem Server, dessen Inhalt und Zustand über mehrere Sessions erhalten bleiben (mit einem Wort: Waren verschwinden nicht aus dem Warenkorb: Der Kunde findet diese Waren bei erneutem Login wieder in seinem Warenkorb vor).

Sessions, die über eine gewisse Zeit hinaus inaktiv waren (d. h. es wurde keine Information in dieser Session abgerufen oder eine Operation ausgeführt), werden aus Sicherheitsgründen ungültig.

Der Kunde kann nach dem Login Artikel im Katalog heraussuchen und in den Warenkorb legen, oder wieder aus dem Warenkorb nehmen (und zwar einzeln oder gleich alle Artikel). Dass sich ein Artikel im Warenkorb befindet, bedeutet an sich noch keine Bestellung. Bestellt wird erst im „Checkout“ (im Englischen soviel wie „den Wagen durch die Kasse schieben“).

4.2 Bestellung und Lieferung

Beim „Checkout“ sieht der Benutzer die AGB, und muss nochmals explizit zustimmen. Dann wird eine Bestellung generiert und der Warenkorb geleert. Die Bestellung wird in das Liefersystem eingespeist. Aufgabe des Liefersystems ist es, die offenen Bestellungen zu verwalten, im Lager die Auslieferung zu veranlassen, den Stand der Zustellung zu überwachen und bei erfolgreicher Zustellung eine Rechnung zu generieren.

Die Lieferabteilung kann offene Bestellungen einsehen (in der Reihenfolge des Eingangs), sucht die nötigen Artikel aus dem Lager und verpackt sie zu einer Sendung auf eine Palette.

[2] B2B steht für *Business to Business*.

Diese wird als Sendung im System registriert, einmal um ein Adressetikett zu erstellen, zum anderen, um die Zustellung der Sendung durch die Spedition zu überwachen.

Bei der Übergabe an die Spedition erhält die Sendung eine Tracking-ID, unter der die Spedition den Lieferstatus zurückmeldet. Sobald die Spedition die Sendung als zugestellt gemeldet hat, wird dies in der Datenbank festgehalten. Erst dann wird eine Rechnung an den Kunden erzeugt, in der Rechnungsabteilung ausgedruckt und an den Kunden geschickt.

4.3 Besteller und Firmen

Nicht jeder, der etwas bestellt, ist deshalb mit einer Firma gleichzusetzen. Großkunden haben häufig mehr als eine Abteilung, welche an den Produkten von *United Clothes* interessiert sein könnte (z. B. die Abteilung für Herrenanzüge und die für Bademoden). Häufig sollen diese Abteilungen unabhängig voneinander nachbestellen können.

Die ins Auge gefasste Lösung besteht darin, jeder Abteilung ein eigenes Login zuzuteilen, jede Abteilung tritt also – im obigen Sinn – als eigener Kunde auf. Danach richtet sich auch die Lieferadresse, da diese einmal mit dem Abteilungsnamen versehen sein muss, andererseits auch einige Textilhäuser mehrere Geschäfte in großen Städten betreiben. Die Rechnung geht andererseits immer an die Buchhaltung der jeweiligen Firma.

4.4 Bestellungen und Verpackungseinheiten

Die Form einer Bestellung bedarf noch einer genaueren Erläuterung. Zu jeder Bestellung müsse sich die Rechnungssumme, Versandgebühren und Mehrwertsteueranteil feststellen lassen. Weiterhin benötigt jede Bestellung einen Vermerk, dass diese Bestellung online abgegeben wurde, den Namen des Betreffenden, Uhrzeit und Datum. Jede Bestellung enthält eine Liste mit Posten, in denen jeweils eine gewisse Anzahl bestimmter Verpackungseinheiten eines Artikels bestellt wird. Beispielsweise:

Anzahl	Artikelnummer	Beschreibung	Verpackungseinheit
12	59837293	schw. T-Shirts X	6er
7	86528294	Krawatten, Raute	10er

Jeder Artikel kann nur in bestimmten Verpackungsformen bzw. -einheiten bestellt werden. Danach richtet sich jeweils der Mengenrabatt (auf den Einzelpreis). Zum Beipiel können T-Shirts einzeln und in 6er-Packs bestellt werden. Im 6er-Pack gibt es 5% Mengenrabatt, d. h. 12 6er-Packs kosten: $12 * 6 * 0.95 * \textit{Einzelpreis}$.

4.5 Aufgabestellung

Erstelle nun ein statisches Modell dieses Web-Shops. Dieses statische Modell soll die „legalen“ Zustände des (intakten) Web-Shop-Systems zu jedem beliebigen Zeitpunkt charakterisieren. Dynamische Aspekte sollen, soweit sinnvoll, in der oben beschriebenen Art annotiert werden.

Lasst Euch übrigens von der gegebenen Beschreibung nicht ablenken: Diese Beschreibung zeigt *Abläufe*. Euer Aufgabe ist es dagegen, zu beantworten, welche Daten, welches Ensemble von Objekten, in dem Shop-System existieren und in welcher Beziehung sie *zu einem bestimmten Zeitpunkt* stehen können.

Weitere Fragen zu Struktur des Shops und zu den Abläufen im Shop beantworte ich gerne auf der Mailingliste. Bitte berücksichtigt, dass ich nicht jede Stunde Mail lese, und nicht garantiere, dass ich Mittwoch abend bis Mitternacht online bin (ziemlich sicher nicht).

Abzugeben sind: Ein Klassendiagramm des Datenmodells des Web-Shops, mit den Anmerkungen zu den Einschränkungen. Handschriftliches ist akzeptabel, sollte aber übersichtlich (nicht zuviel ausgestrichen) und verständlich bleiben.

Viel Spaß!