Software Engineering I

Musterlösungen zur Klausur vom 2.8.2003

Aufgabe 1

a) In der Aufgabenstellung war ein möglichst einfaches Klassendiagramm gefordert. Daher verzichten wir auf Klassen wie Sofortkauf, Bewertung oder Benachrichtigung. Abb. 1 zeigt eine mögliche Lösung.

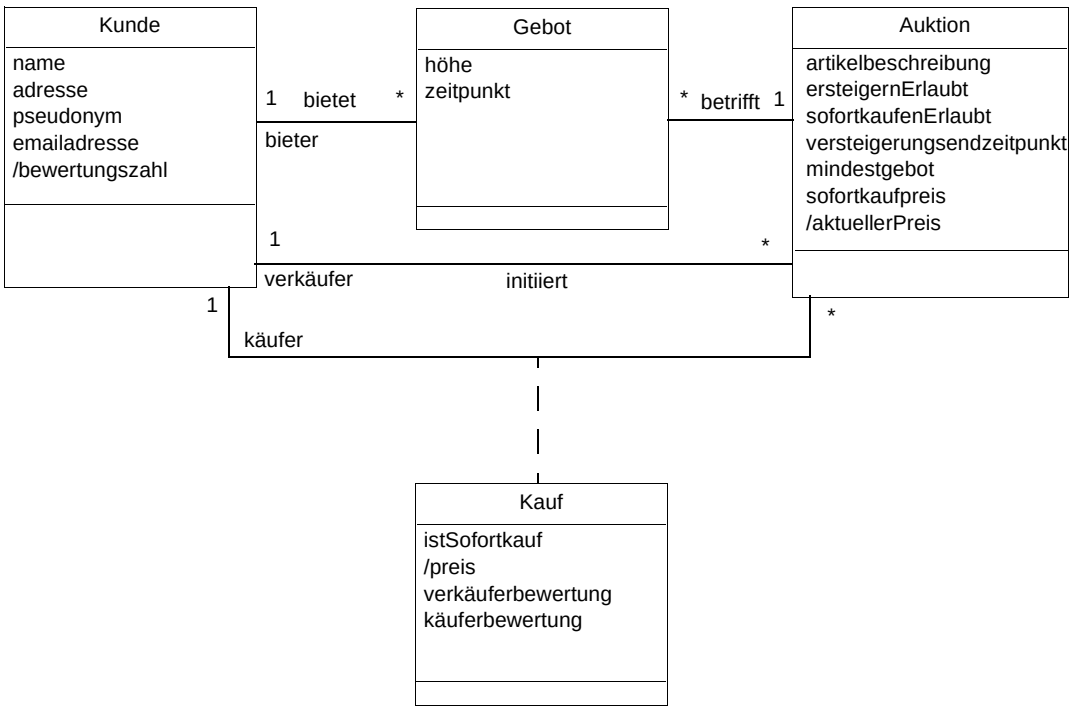

Abb. 1: Klassendiagramm des Online-Auktionssystems

Ein Kunde kann als Verkäufer eine Auktion initiieren, als Käufer einen Kauf tätigen und als Bieter ein Gebot abgeben. Im ersten Fall ist keine Zusatzinformation notwendig, daher wird eine einfache Assoziation initiiert modelliert. Im zweiten und dritten Fall ist Zusatzinformation notwendig, die durch Attribute der Klassen Kauf und Gebot modelliert wird. Die Realisierung dieser Klassen ist jedoch unterschiedlich, weil es zu jedem Paar (Kunde, Auktion) nur eine Kauf-Instanz, aber mehrere Gebot-Instanzen geben kann. Da zwei Objekte über eine Assoziation nicht mehrfach verbunden sein können, kann nur die Klasse Kauf als Assoziationsklasse modelliert werden, während Gebot eine eigenständige Klasse sein muss.

b) Abb. 2 zeigt ein Objektdiagramm des Online-Auktionssystems.

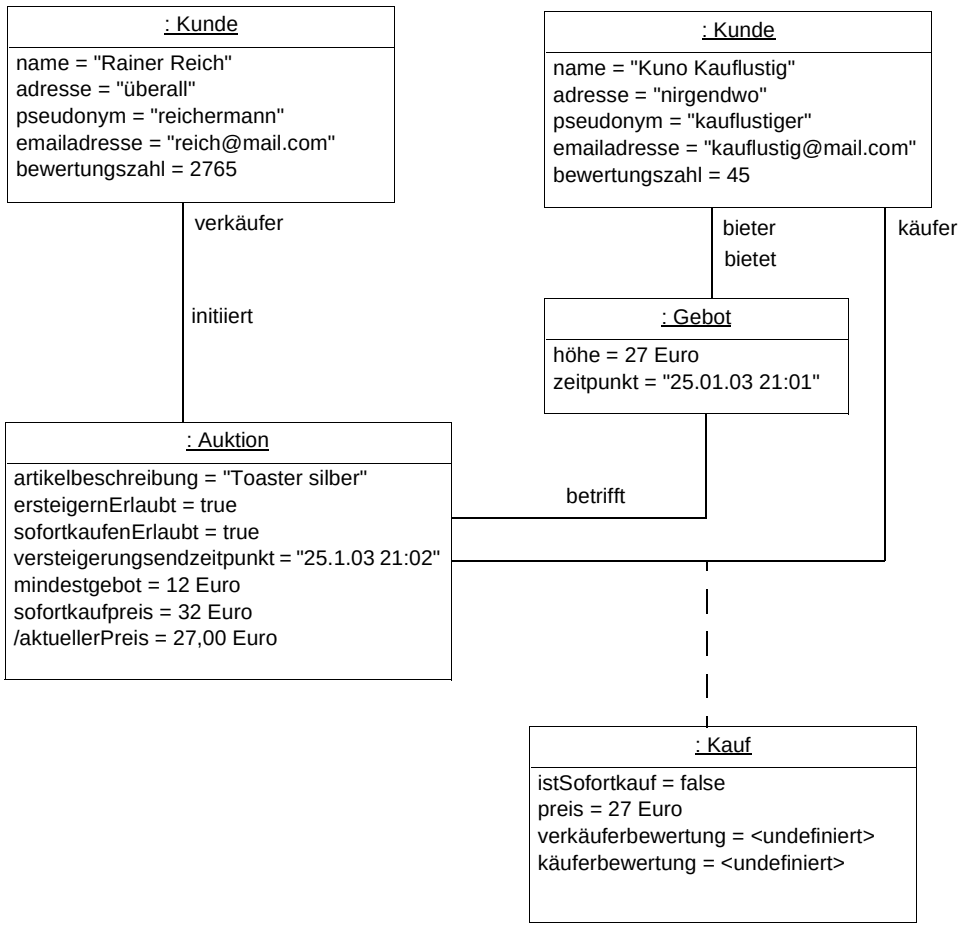

Abb. 2: Objektdiagramm des Online-Auktionssystems

c) Anwendungsfallspezifikation für den Anwendungsfall "Auktion beenden":

use case Auktion beenden

actors
Systemuhr

precondition
Bei einer Auktion wurde das Versteigerungs-Endzeitpunkt überschritten.

main flow
Das System ermittelt das höchste Gebot.
Das System registriert einen Kauf des versteigerten Artikels durch den Höchstbietenden.
Der Verkäufer und alle Bieter werden über den Ausgang der Versteigerung informiert.

postcondition
Für die Auktion können keine Gebote mehr abgegeben werden.

exceptional flow keine Gebote vorhanden
Der Verkäufer wird informiert, dass der Artikel nicht ersteigert wurde.

postcondition
Für die Auktion können keine Gebote mehr abgegeben werden.

end Auktion beenden.

Aufgabe 2

Abb. 3 zeigt ein Zustandsdiagramm des mobilen CD-Spielers.

Alternativ ist es möglich, nur den Zustand eines eingeschalteten CD-Spielers zu modellieren: In diesem Fall fällt der Zustand Ausgeschaltet weg, vom Initialzustand aus wird direkt der Zustand Wiedergabe erreicht, und beim Ausschalten der Terminalzustand.

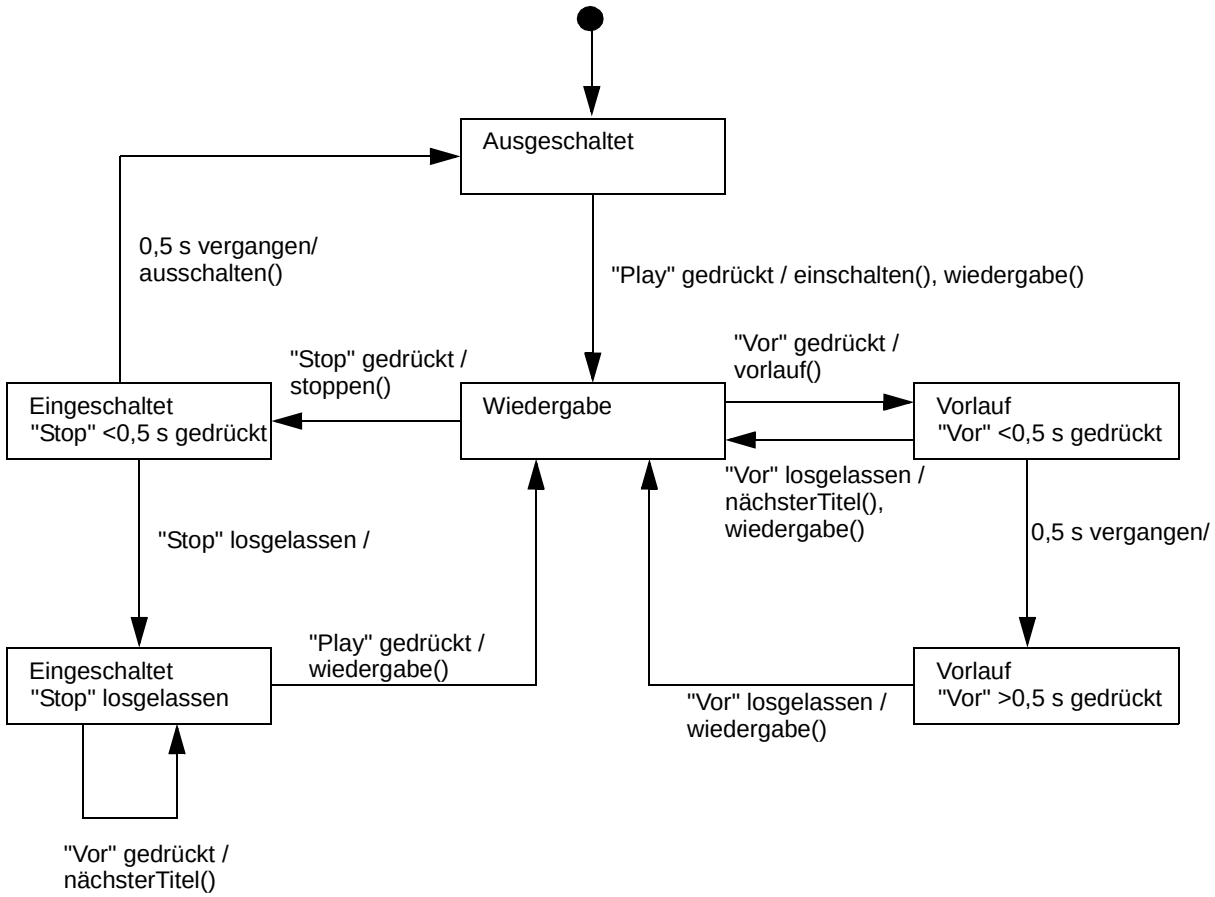

Abb. 3: Zustandsdiagramm für den mobilen CD-Spieler

Aufgabe 3

a) Ein Iterator ist ein Objekt, das die Elemente eines Behälterobjekts aufzählen kann und zu diesem Zweck die Operationen hasNext():boolean und next():Object anbietet.

b) Abb. 4 zeigt das Sequenzdiagramm für den Ablauf der Operation toString() der Klasse HashSet.

c) Die Variable index hat den Wert 1, weil an der Indexposition 1 das erste gültige Element liegt.

d) Die Operation muss nur einmal aufgerufen werden, weil sie das nächste *verwendete* (=gültige) Element sucht.

e) Abb. 5 zeigt das verfeinerte Sequenzdiagramm für den Ablauf der Operation toString() der Klasse HashSet.

f) Wenn dem HashSet neue Elemente hinzugefügt werden, werden diese an irgendwelchen (von der Hashfunktion abhängigen) Stellen in das Array hashtable eingefügt. Der Iterator wird die neu eingefügten Elemente ausgeben, sofern sie im Array hinter der Stelle index liegen. Mit anderen Worten: Die nachträglich eingefügten Elemente werden "vielleicht" ausgegeben.

Anmerkung:

Eine gute Iterator-Implementierung sollte im Fall von nachträglichen Veränderungen der durchlaufenen Datenstruktur entweder eine Fehlermeldung liefern oder ein anderes, klar definiertes Verhalten zeigen.

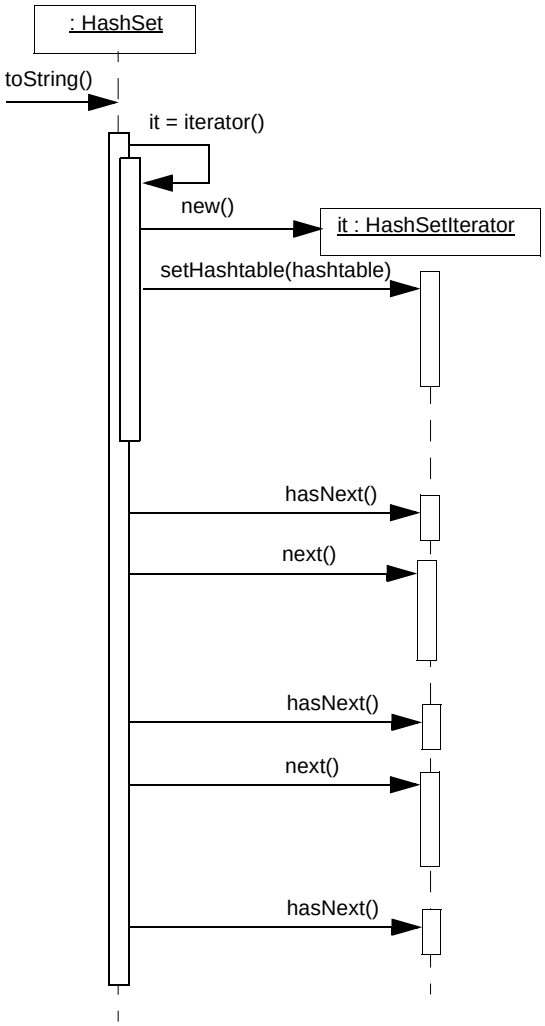

Abb. 4: Sequenzdiagramm der Operation toString() der Klasse HashSet

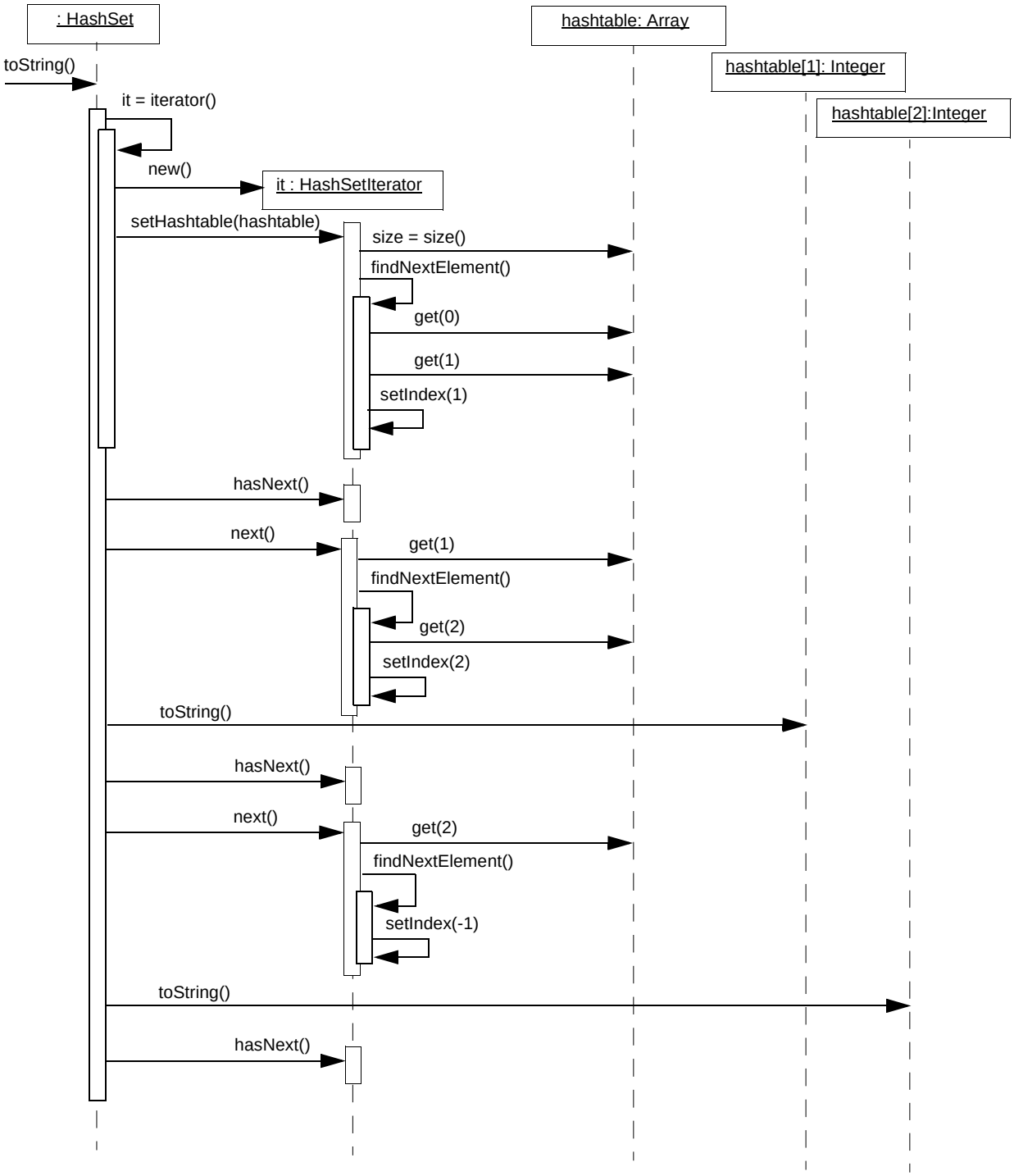

Abb. 5: Verfeinertes Sequenzdiagramm der Operation toString() der Klasse HashSet

Aufgabe 4

Abb. 6 zeigt das durch die Transformation entstehende Feinentwurfs-Klassendiagramm.

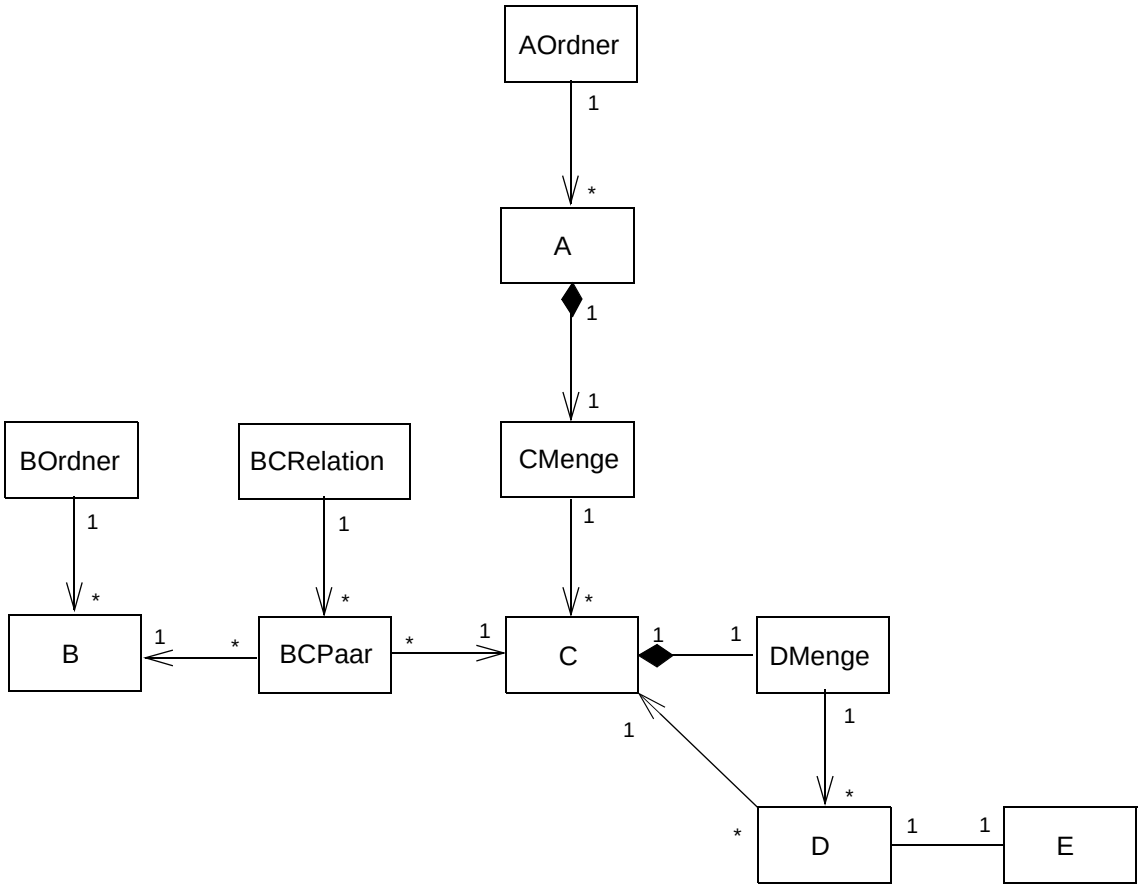

Abb. 6 Das transformierte Klassendiagramm

Aufgabe 5

a) Es sind fünfVerbindungen erforderlich, vgl. Abb. 7.

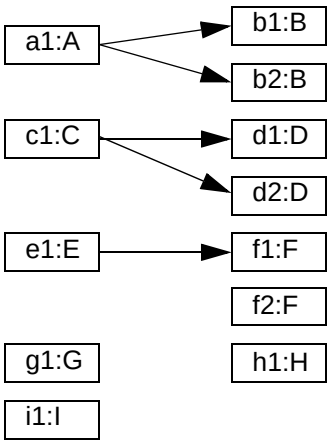

Abb. 7: Das Objektdiagramm mit minimaler Anzahl von Verbindungen

b) Die Objekte b1, b2, d1, d2 und f1 sind von Instanzen anderer Klassen aus erreichbar.

c) Von den Klassen B und D ist jede Instanz erreichbar.

d) Die Klassen A, C, E, F, G, H, und I sind Echte Ganzes-Klassen.

e) Der Satz lautet wie folgt:

Eine Klasse K ist keine Echte Ganzes-Klasse, wenn es eine Klasse K’ und eine Assoziation zwischen den Klassen K und K’ gibt, die von K’ nach K navigierbar ist und bei der die untere Grenze der Multiplizität am K’-Ende größer Null ist.

f) Echte Ganzes-Klassen sind genau die Klassen, für die im Entwurf auf jeden Fall ein Ordner vorgesehen werden muss, da sonst nicht alle Instanzen erreichbar sind.