INSTITUT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Prof. Dr. Dr. h.c. M. Broy
Dr. H. Ehler, S. Wagner

Lösungsblatt 5
23. November 2006

Übungen zu Softwaretechnik

Aufgabe 8 – Zustandsmodellierung eines Bankomat

Gegeben sei ein Geldautomat, wie er bei Banken üblicherweise zu finden ist. Dieser soll als formales Zustandsübergangsdiagramm mit Ein- und Ausgabe, Vor- und Nachbedingungen modelliert werden. Zu modellieren sind dabei:

- *die Authentifizierung mittels PIN-Eingabe,*
- *die Auszahlung eines gewünschten Geldbetrags,*
- *ein maximaler Auszahlungsbetrag von 500,- Euro,*
- *der Einzug der Kundenkarte nach dreimalig falscher Eingabe der PIN,*
- *Abbruch in jedem Zustand möglich.*

Folgende Hilfsfunktionen können als gegeben betrachtet werden:

- *check_PIN(pin : Int) : Bool prüft ob es sich um die korrekte PIN handelt.*
- *gibGeldscheine(betrag: Int) : void gibt die Geldscheine an den Benutzer aus.*

Nicht berücksichtigt werden müssen weitere Menüoptionen wie Kontostandsabfrage und Überweisungen sowie die Stückelung des Geldes in Scheine und Münzen.

a) Geben Sie die Ein- und Ausgaben des Geldautomaten an.

Eingaben: Karte, Geldbetrag, PIN (oder Nat, Zahlen, Integer), Abbruch

Ausgaben: Karte, Geld, Bildschirmausgaben (z.B. PIN eingeben, PIN falsch, Karte entnehmen, Betrag zu hoch)

b) Geben Sie alle Zustandsattribute an, die für die Modellierung des Geldautomaten notwendig sind und beschreiben Sie deren Verwendungszweck.

a: Integer (+1) Anzahl der PIN-Eingabeversuche

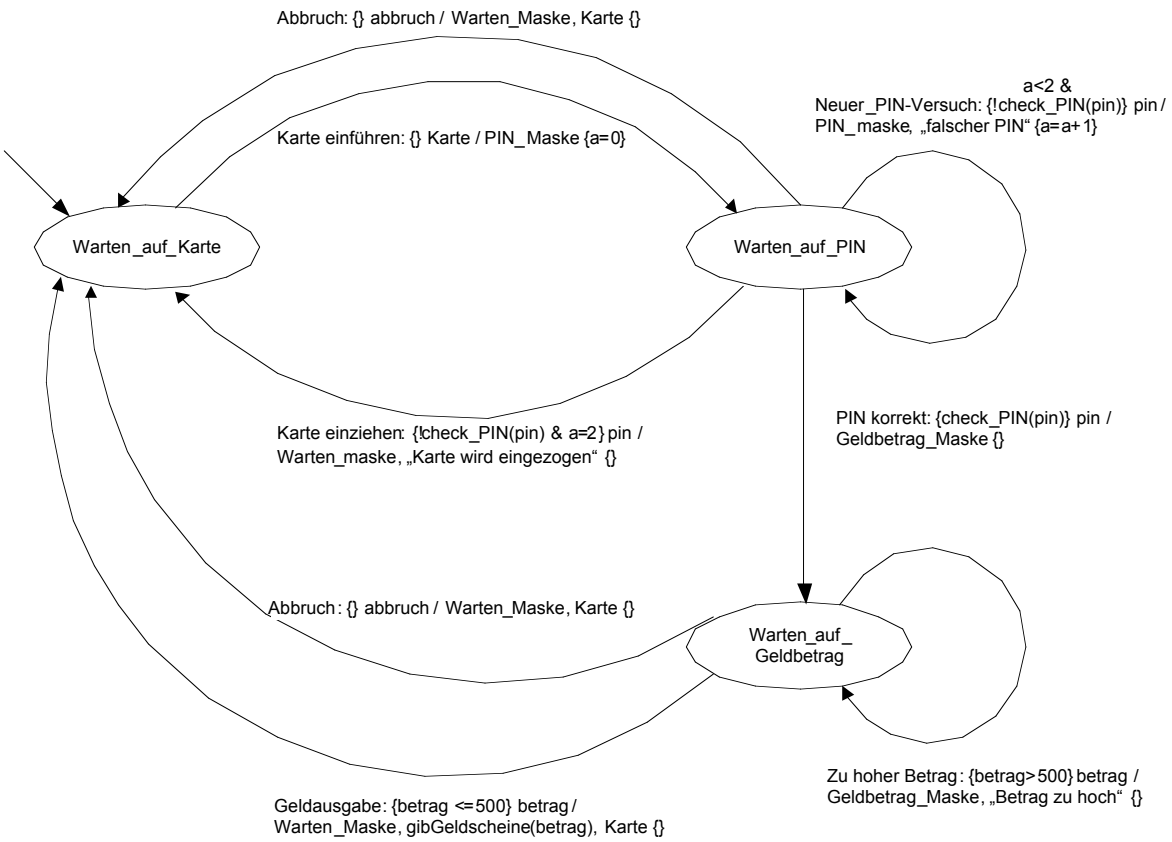
c) Identifizieren Sie anhand der Zustandsattribute die Zustände des Geldautomaten und geben Sie eine Charakterisierung der Zustände durch Angabe der möglichen Wertebereiche der Zustandsattribute an. Welcher der Zustände ist Anfangszustand?

Warten_auf_Karte: keine Karte im Automat, ist Anfangszustand

Warten_auf_PIN: Karte eingeführt, PIN muss eingegeben werden

Warten_auf_ Betrag: Karte im Automat, PIN wurde korrekt eingegeben, Betrag wird erwartet

d) Zeichnen Sie das Zustandsübergangsdiagramm. Verwenden Sie hierzu die Syntax mit Ein- und Ausgabe, Vor- und Nachbedingungen.


Abbruch: {} abbruch / Warten_Maske, Karte {}
a<2 &
Neuer_PIN-Versuch: {!check_PIN(pin)} pin /
PIN_maske, „falscher PIN“ {a=a+1}
Karte einführen: {} Karte / PIN_Maske {a=0}
Warten_auf_Karte
Warten_auf_PIN
Karte einziehen: {!check_PIN(pin) & a=2} pin /
Warten_maske, „Karte wird eingezogen“ {}
PIN korrekt: {check_PIN(pin)} pin /
Geldbetrag_Maske {}
Abbruch: {} abbruch / Warten_Maske, Karte {}
Warten_auf_
Geldbetrag
Zu hoher Betrag: {betrag>500} betrag /
Geldbetrag_Maske, „Betrag zu hoch“ {}
Geldausgabe: {betrag <=500} betrag /
Warten_Maske, gibGeldscheine(betrag), Karte {}

Aufgabe 9 – Zustandsmodellierung der Lagerverwaltung

In dem bisher betrachteten Beispiel eines betrieblichen Informationssystems soll das Verhalten des Systems modelliert werden. Dazu wollen wir das Verhalten eines Produkts im System modellieren. Als zusätzliche Restriktion wollen wir einführen, dass ein Produkt erst dann bestellt wird, wenn keine entsprechenden Produkte mehr auf Lager sind. Dadurch verändern Einkäufe, Verkäufe und Bestellungen den Zustand eines Produkts.

a) Beschreiben Sie ein Produkt durch einen Zustandsautomaten.

Die Menge der Zustände eines Produktes prod ist nach dem Datenmodell gegeben durch die beiden Attribute:

- onstock (Anzahl der Produkte, die auf Lager sind)
- ordered (Anzahl der bestellten Produkte).

Zustandsmenge ist also $IN \times IN$.

Die möglichen Eingaben sind die Prozeduraufrufe:

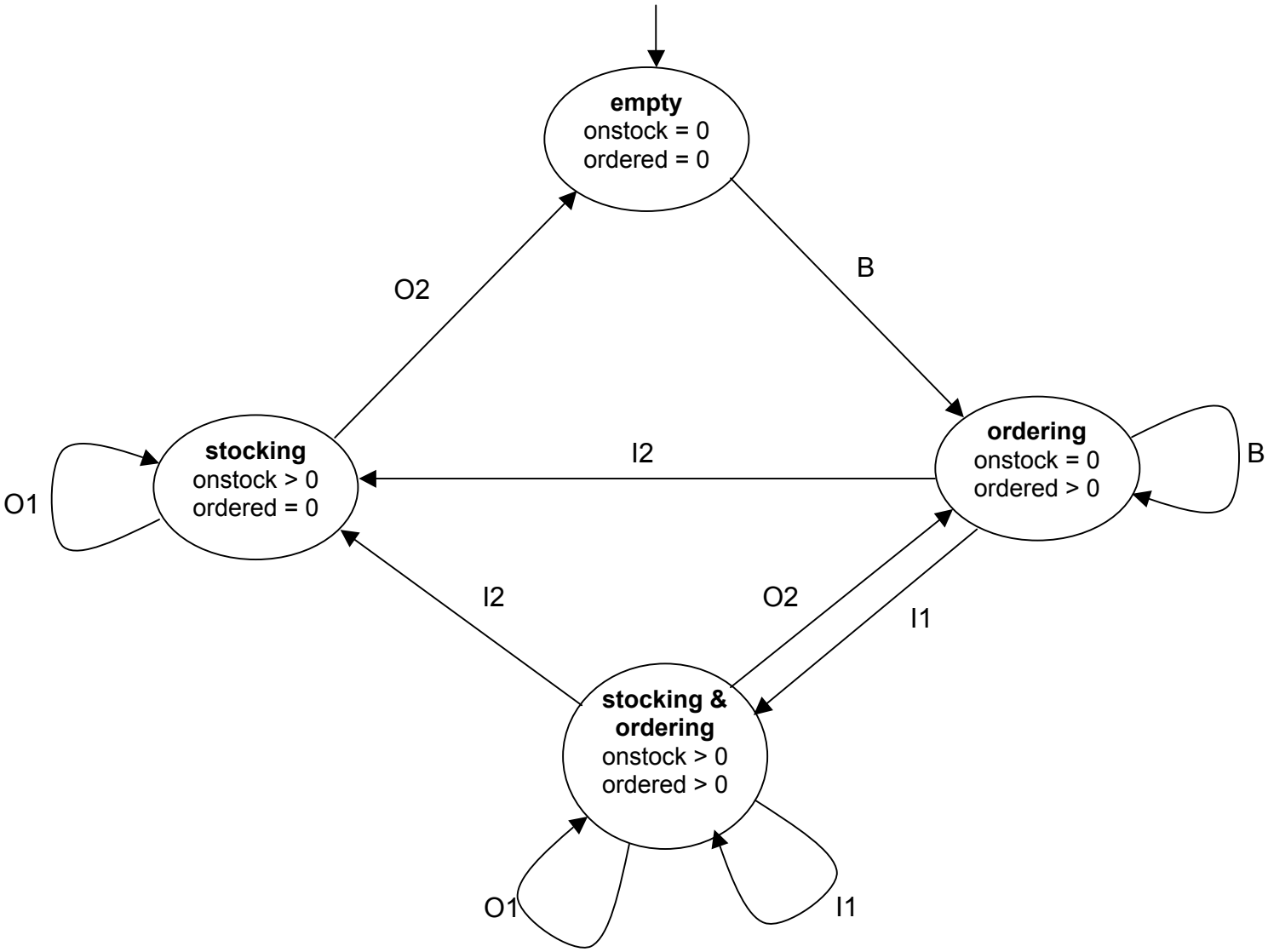
- prod_order(prod, ord), hier abgekürzt zu order(ord)
- prod_outgoing(prod, out), hier abgekürzt zu outgoing(out)
- prod_incoming(prod, in), hier abgekürzt zu incoming(in)

Die Ausgaben der Prozeduren sind nur eine Meldung der Terminierung ack (acknowledgement).

Die Beschriftung der möglichen Zustandsübergänge kürzen wir folgendermaßen ab:

- Bestellung (d.h. Eingabe order(ord)):
  B: {} order(ord) / ack {ordered = ordered' + ord}
- Verkauf (d.h. Eingabe outgoing(out)):
  O1: {out < onstock} outgoing(out) / ack {onstock = onstock' - out}
  O2: {out = onstock} outgoing(out) / ack {onstock = onstock' - out}
- Einkauf (d.h. Eingabe incoming(in)):
  I1: {in < ordered} incoming(in) / ack { onstock = onstock' + in ∧
  ordered = ordered' - in }
  I2: {in = ordered} incoming(in) / ack { onstock = onstock' + in ∧
  ordered = ordered' - in }

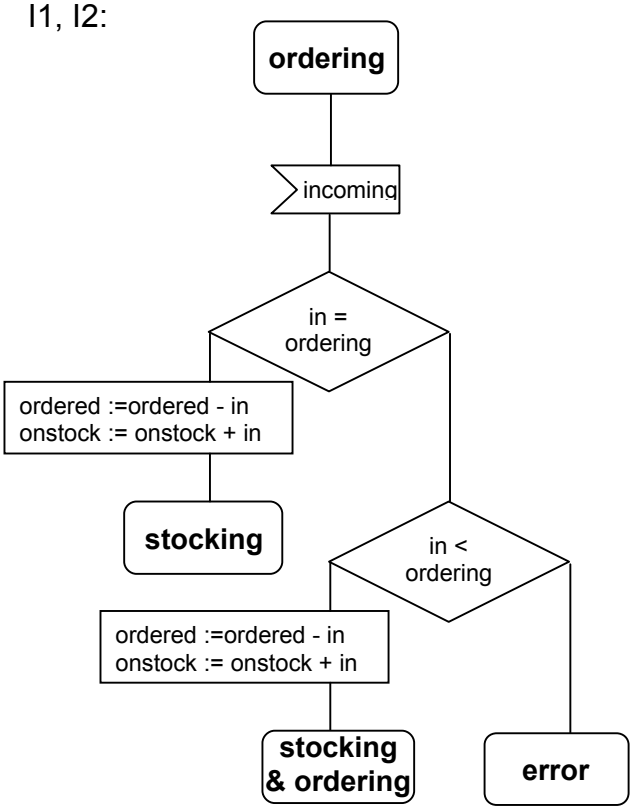
Damit erhalten wir folgendes Zustandübergangsdiagramm:

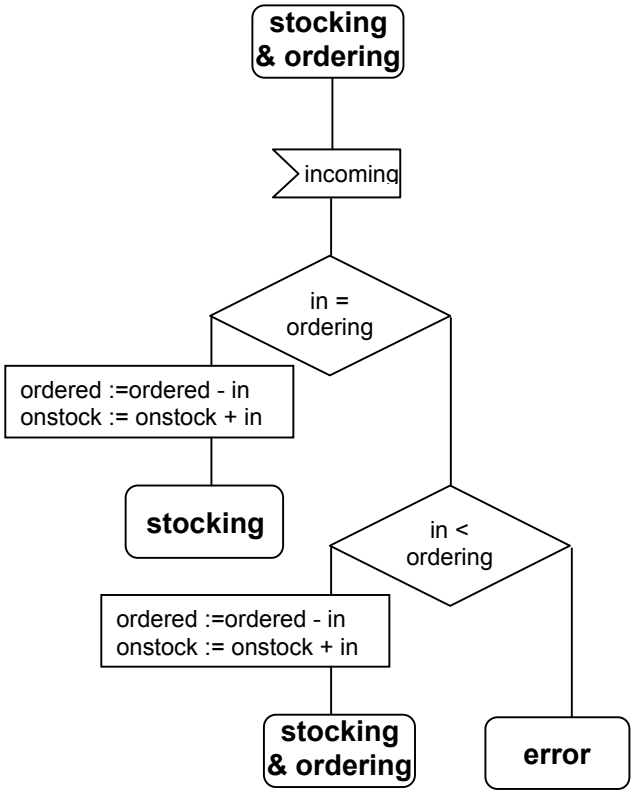


b) Beschreiben Sie das Produkt durch ein Kontrollflussdiagramm nach SDL.

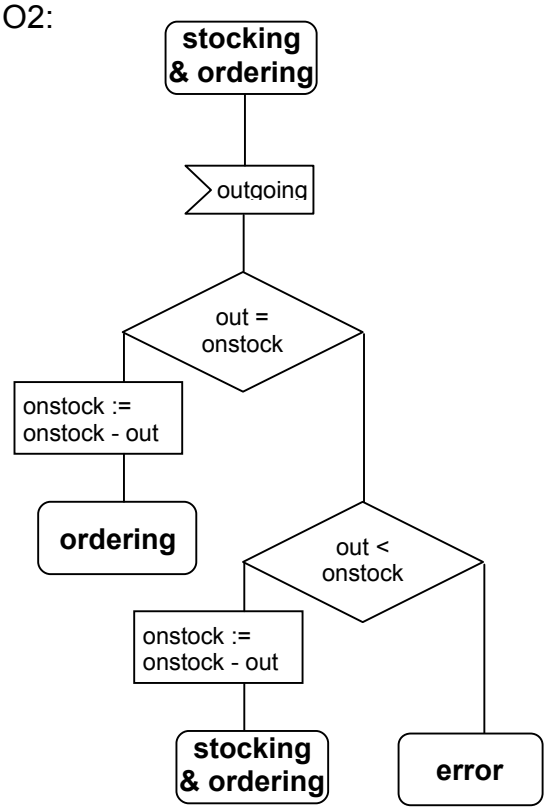
B:

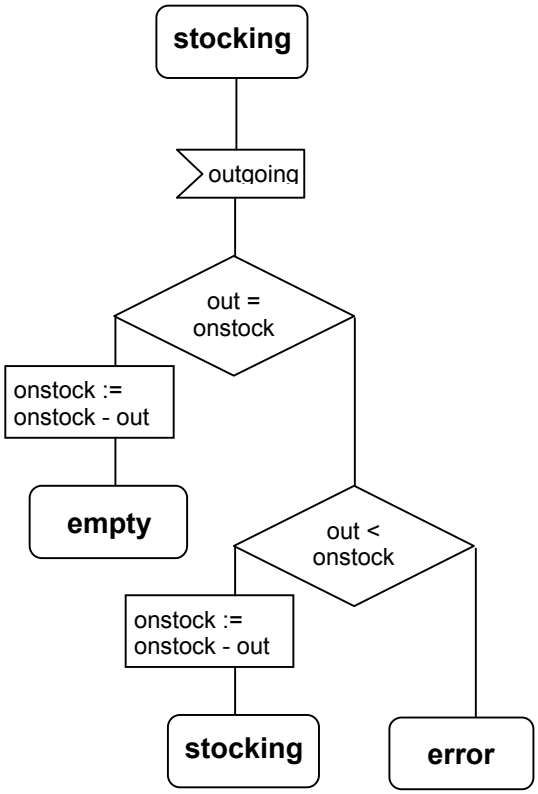
I1, I2:


ordering
incoming
in =
ordering
ordered :=ordered - in
onstock := onstock + in
stocking
in <
ordering
ordered :=ordered - in
onstock := onstock + in
stocking
& ordering
error


stocking
& ordering
incoming
in =
ordering
ordered :=ordered - in
onstock := onstock + in
stocking
in <
ordering
ordered :=ordered - in
onstock := onstock + in
stocking
& ordering
error

O1, O2:


stocking
& ordering
outgoing
out =
onstock
onstock :=
onstock - out
ordering
out <
onstock
onstock :=
onstock - out
stocking
& ordering
error


stocking
outgoing
out =
onstock
onstock :=
onstock - out
empty
out <
onstock
onstock :=
onstock - out
stocking
error