Technische Universität Berlin
Institut für Softwaretechnik und Theoretische Informatik
FG Softwaretechnik
Franklinstr. 28/29
10587 Berlin

Jähnichen
Kammüller
Sokenou

Klausur

Softwaretechnik

25. Juli 2001

Prüfen Sie zunächst, ob dieses Exemplar vollständig ist (13 Blätter ohne Deckblatt).

Tragen Sie auf diesem Deckblatt und darüber hinaus auf allen Blättern, die Sie für Ihre Niederschrift verwenden, Ihren Namen und Ihre Matrikelnummer ein.

Zum Bestehen der Klausur sind **35 Punkte** notwendig. Insgesamt sind 95 Punkte möglich. Die Bearbeitungszeit beträgt 120 Minuten. In der Klausur sind **außer Telekommunikationseinrichtungen sämtliche Hilfsmittel** zugelassen.

Viel Erfolg!

Name, Vorname: ______________________________

Matrikelnummer: ______________

Studienrichtung/Prüfungsordnung: ______________
SWT-Übungen habe ich im Semester ________besucht.

Aufgabe	1	2	3	Gesamt
Maximal:	40	35	20	95
Erreicht:				

Aufgabe 1 (Fusion) - Die Videothek

(Geschätzte Bearbeitungszeit: 50 min.)

In einer Videothek können gegen Rechnung *Filme* ausgeliehen werden. Dabei kann es zu jedem Film mehrere ausleihbare *Kopien* geben.

Ein neuer Film kann mit einer Kopie von einem *Mitarbeiter* der Videothek in das System *aufgenommen* oder wieder *entfernt* werden. Dabei muß zu jedem Film *mindestens* eine Kopie existieren. Ebenso ist es möglich, weitere Kopien eines Films aufzunehmen oder zu entfernen. Jeder Film vermerkt die *Anzahl seiner Kopien*. Der Bestand enthält das Filmangebot und vermerkt die *Anzahl der Filme* (*nicht* die Anzahl der Kopien!).

Die Kopien stehen in *Regalen*. Verschiedene Kopien eines Films können in unterschiedlichen Regalen stehen. Handelt es sich um einen ein Actionfilm, so könnte z. B. eine Kopie im Regal für Neuheiten als auch eine Kopie im Actionfilm-Regal stehen.

Ein *Kunde leiht* einen Film, indem er zunächst seine Kundenkarte und anschließend die aus einem Regal entnommene Kopie unter einen Barcode-Scanner hält. Der Barcode auf der Hülle der Kopie enthält die eindeutige Kopie-Nummer.

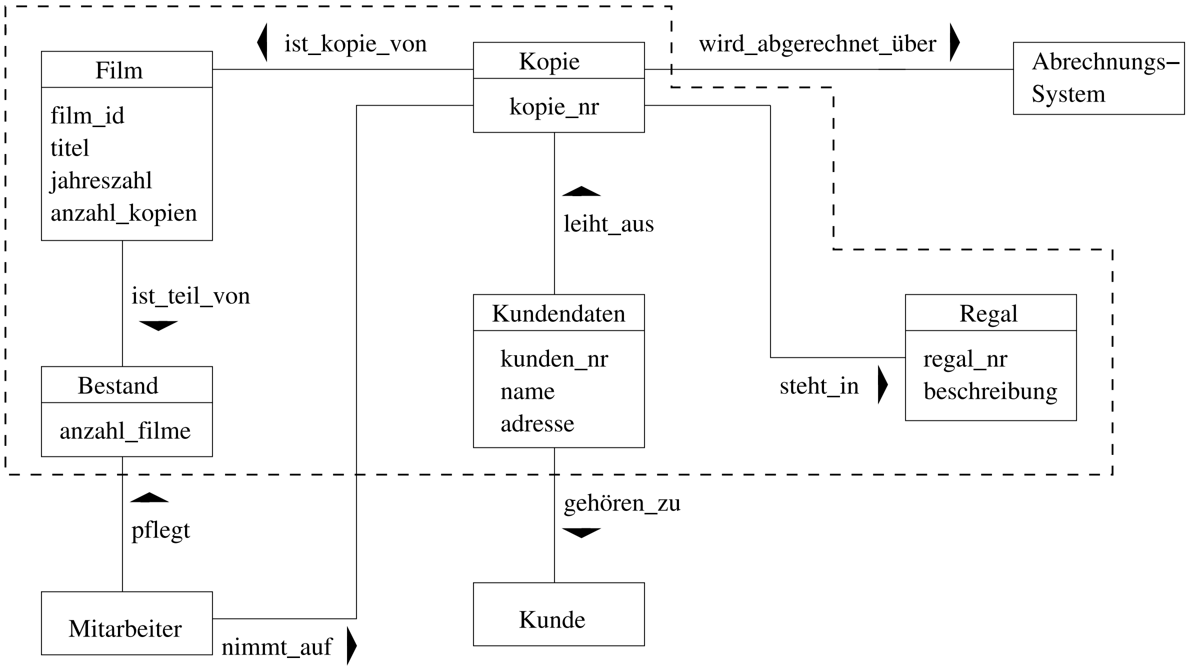
Die Abrechnung erfolgt durch ein unabhängiges *Abrechnungssystem*. Dafür ist es nötig, bei Ausleihe sowie Rückgabe der Kopie die Daten an dieses zu übermitteln.

Aufgabenstellung

Die Entwicklung des Sytems soll mit Hilfe der Fusion-Methode durchgeführt werden. Insbesondere sollen die Funktionalitäten *kopie_ausleihen* und *kopie_entfernen* betrachtet werden.

1. Geben Sie die *Multiplizitäten* im gegebenen Systemklassenmodell an.
   Geben Sie explizit an, wenn ein Objekt mit beliebig vielen anderen Objekten in Beziehung steht. *5 Punkte*
2. Benennen Sie die *Akteure*, die mit dem System kommunizieren. *3 Punkte*
3. Vervollständigen Sie die *Operationsschemata* für die Systemoperationen *kopie_ausleihen* und *kopie_entfernen*.
   Berücksichtigen Sie dabei **keine** Fehlerfälle!
   Geben Sie **nur** für die Operation *kopie_entfernen* die Nachbedingung an. *15 Punkte*
4. Stellen Sie die *Objektinteraktionsgraphen* für die Systemoperationen *kopie_ausleihen* und *kopie_entfernen* auf. *15 Punkte*
5. Nennen Sie eine *Konsistenzbedingung* zwischen den Modellen, die im Fusionprozeß entwickelt werden. *2 Punkte*

1. Systemklassenmodell



2. Akteure

3. Operationsschemata

Op.	=	kopie_ausleihen
Desc.	=	Ein Kunde leiht eine Kopie aus, indem er per Barcode-Scanner Kundennummer und Kopienummer eingibt.
Input	=	kopie_nr, kunden_nr
Reads	=	
Changes	=	
Sends	=	
Pre	=	$\forall\, kd : Kundendaten \bullet (kd, kopie) \notin leiht\_aus$
Post	=	*Braucht nicht angegeben zu werden!*

Op.	=	kopie_entfernen
Desc.	=	Ein Mitarbeiter entfernt eine Kopie eines Films.
Input	=	film_id, kopie_nr
Reads	=	
Changes	=	
Sends	=	
Pre	=	$\forall\, kd : Kundendaten \bullet (kd, kopie) \notin leiht\_aus$
Post	=	

4. Objektinteraktionsgraphen

5. Konsistenzbedingung

Aufgabe 2 (Object-Z) - Offiziersskat

(Geschätzte Bearbeitungszeit: 45 min.)

Spielregeln

Das Spiel ähnelt Skat oder Bridge, ist aber nur für zwei Spieler. Die Spieler heißen S_1 und S_2. Zu Beginn eines Spieles ist immer S_1 an der Reihe. Es gibt 20 Karten, unterschieden durch 4 Farben ($\diamondsuit$, $\heartsuit$, $\spadesuit$, $\clubsuit$) und 5 Werte (in aufsteigender Reihenfolge): Zehn, Bube, Dame, König, As.

Jeder Spieler erhält 10 Karten auf die *Hand*, die im Spiel nacheinander abwechselnd *ausgespielt* werden: Ein Spieler spielt eine Karte von der Hand aus, der andere *erwidert* (s.u.) durch das Spielen einer Karte von seiner Hand. Karten werden in einem Spiel nur einmal gespielt, d.h. pro Runde verringert sich die Anzahl der Karten auf der Hand der Spieler um eins. Pro Spiel gibt es 10 Runden.

Eine der beiden Karten ist je nach Farbe, Wert, oder je nachdem, ob sie zuerst gespielt wurde, stärker als die andere – das heißt, sie *sticht* die andere. Derjenige Spieler, der so gestochen hat, erhält einen Punkt (oder *Stich*) und ist als nächster *an der Reihe* auszuspielen. Der Spieler, der an der Reihe ist, spielt eine beliebige Karte aus. Der zweite Spieler erwidert das Ausspielen. Es gibt drei Möglichkeiten auf eine ausgespielte Karte zu *erwidern*:

1. Hat der erwidernde Spieler noch eine Karte auf der Hand, die die *gleiche Farbe* hat, wie die ausgespielte Karte, so muß er *bekennen*, d.h. eine seiner Karten mit gleicher Farbe spielen. Es sticht die Karte mit höherem Wert.
2. Hat er keine Karte mit gleicher Farbe mehr, so kann er entweder *trumpfen*, d.h. eine Karte spielen, deren Farbe Trumpf ist,
3. oder eine Karte anderer Farbe *abwerfen*. Durch Trumpfen wird gestochen, durch Abwerfen nicht (der Stich geht an den Ausspielenden).

Gewonnen hat der Spieler, der am Ende mehr Stiche hat.

Aufgabenstellung

Spezifizieren Sie das oben beschriebene Kartenspiel in Object-Z. Benutzen Sie dazu die Vorgaben.

Hinweis: Beachten Sie dabei die Abhängigkeiten zwischen den Klassen $Spiel$ und $Spieler$, sowie die Tatsache, daß es 2 Spieler gibt!

1. Spieler

Vorgegeben sind die Typdefinitionen für Werte, Farben, und Karten, die Funktionen *farbe* und *wert*, sowie die Relation $\prec$ auf den Werten ($x \prec y \Leftrightarrow$ der Wert von y ist höher als der von x).

Spezifizieren Sie die Klasse Spieler:

1. Vervollständigen Sie die Operationen *ausspielen*, *trumpfen*, *abwerfen* und *bekennen* durch Eintragen der passenden Prädikate. *18 Punkte*

2. Definieren Sie die Operation *erwidern* durch Verwendung des Schemakalküls. *5 Punkte*

$Wert == Zehn \mid Bube \mid Dame \mid König \mid As$
$\prec \quad == \{(Zehn, Bube), (Bube, Dame), (Dame, König), (König, As)\}^+$
$Farbe == \diamondsuit \mid \heartsuit \mid \spadesuit \mid \clubsuit$
$Karte == Farbe \times Wert$
$farbe == first$
$wert == second$

Spieler

$hand : \mathbb{F}\, Karte$
$trumpf : Farbe$
$anderReihe : \mathbb{B}$
$stiche : (0..10)$

INIT

$stiche = 0$
$trumpf \in Farbe$
$\#hand = 10$

ausspielen

$\Delta(hand, anderReihe, stiche)$
$liegt! : Karte$
$stich\_gewonnen? : \mathbb{B}$ [Stich wird vom ausspielenden Spieler gewonnen]

$trumpfen$

$\Delta(hand, anderReihe, stiche)$
$liegt? : Karte$
$spiele! : Karte$
$stich\_gewonnen! : \mathbb{B}$ [Meldung an den Gegner]

$abwerfen$

$\Delta(hand)$
$liegt? : Karte$
$spiele! : Karte$
$stich\_gewonnen! : \mathbb{B}$ [Meldung an den Gegner]

$bekennen$

$\Delta(hand, anderReihe, stiche)$
$liegt? : Karte$
$spiele! : Karte$
$stich\_gewonnen! : \mathbb{B}$ [Meldung an den Gegner]

$erwidern \mathrel{\widehat{=}}$

2. Spiel

Spezifizieren sie nun das Kartenspiel:

1. Vervollständigen Sie das Zustandsschema und das Initschema — *5 Punkte*
2. Fassen Sie nun den gesamten Ablauf eines Spieles in der Operation *spielen* zusammen, indem Sie den Schemakalkül anwenden. — *7 Punkte*

Spiel

$S\_1, S\_2 : Spieler$

$S\_1\_gewinnt, S\_2\_gewinnt : \mathbb{B}$

$S\_1\_gewinnt \Leftrightarrow$

$S\_2\_gewinnt \Leftrightarrow$

INIT

$spielen \mathrel{\widehat{=}}$

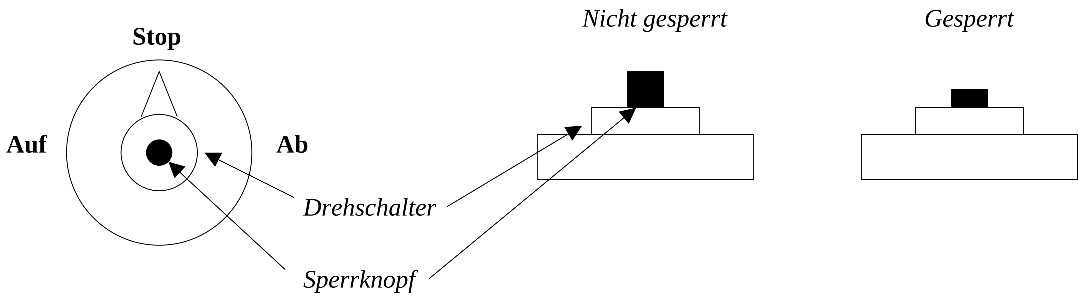
Aufgabe 3 (Statecharts) - Die Jalousiensteuerung

(Geschätzte Bearbeitungszeit: 25 min.)

Durch unsachgemäße Bedienung der Jalousien im TU-Mathegebäude sind diese zum Teil zerstört und unbrauchbar geworden. Die Mitarbeiter in den Räumen blockierten den Bedienschalter, da sie das automatische Herauf- und Herunterfahren der Jalousien bei Sonne störte. So wurde aber auch das automatisches Herauffahren der Jalousien bei starkem Wind verhindert, das die Jalousien vor Beschädigung schützen sollte.

Eine neue Jalousiensteuerung soll die alte Funktionalität erhalten und um eine ungefährliche Möglichkeit zur Blockierung erweitern. Im einzelnen soll folgende Funktionalität unterstützt werden:

- Automatisches Herunterfahren bei sonnigem Wetter, Herauffahren bei bedecktem Himmel.
- Automatisches Herauffahren bei Wind. Um nicht bei jeder einzelnen Windboe zu reagieren, wird das Herauffahren erst nach 2 Sekunden Wind ausgelöst.
- Bedienung durch die Mitarbeiter mit Hilfe eines Bedienschalters (siehe Bild). Wird der Schalter z. B. in die Position "Auf" gebracht, läuft der Motor entweder bis gestoppt wird oder 30 Sekunden lang (wobei die Position der Jalousie keine Rolle spielt). Das Herunterfahren ist analog.
- Zusätzlich wird eine Sperre in Form eines Knopfes eingeführt, der hereingedrückt oder gelöst werden kann (siehe Bild), so daß ein ständiges Herauf- und Herunterfahren der Jalousie bei Sonne verhindert wird. Die Sperre wird automatisch bei Wind gelöst, so daß die Jalousie herauffahren kann.



Aufgabenstellung

Modellieren Sie die Jalousiensteuerung gemäß den oben beschriebenen Anforderungen mit Hilfe von Statecharts.

1. Geben Sie die externen Ereignisse an, auf die das System reagiert und dokumentieren Sie diese *kurz*. *3 Punkte*
2. Geben sie den Zustandsautomaten inklusive all seiner Teile an. *14 Punkte*
3. Geben Sie die internen Ereignisse an, die durch das System erzeugt werden und dokumentieren Sie diese *kurz*. Falls Sie keine internen Ereignisse verwenden, begründen Sie *kurz*, warum diese in Ihrer Modellierung nicht notwendig sind. *3 Punkte*

Hinweis: Es ist nicht nötig, den Schalter bzw. die Schalterstellungen zu modellieren!

1. Externe Ereignisse

3. Interne Ereignisse

2. Statechart Jalousiensteuerung

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel