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

Kammüller
Jucknath
Mosconi
Oeters

Klausur

Softwaretechnik

Wintersemester 03/04, 25. Februar 2004

Prüfen Sie zunächst, ob dieses Exemplar vollständig ist (25 Blätter ohne Deckblatt). Bitte keine eigenen Blätter verwenden.

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 100 Punkte möglich. Die Bearbeitungszeit beträgt 120 Minuten. In der Klausur sind **außer Telekommunikationseinrichtungen sämtliche Hilfsmittel** zugelassen.

Viel Erfolg!

Name, Vorname: ____________________________________

Matrikelnummer: ____________________

Studienrichtung: ____________________

SWT-Übungen habe ich im Semester ____________besucht.

Aufgabe	1	2	3	Gesamt
Maximal:	40	35	25	100
Erreicht:				

Aufgabe 1 (Fusion) - Verwaltung von Veranstaltungen und Prüfungen im Universitätsbetrieb

(Geschätzte Bearbeitungszeit: 50 min.)

Modelliert werden soll ein System zur Verwaltung einer Grundstudiumsvorlesung (beispielsweise Informatik 1 oder Informatik 2) in einem Sekretariat. Wir betrachten für unser System nur ein Sekretariat, von dem aus mehrere Semester dieser Grundstudiumsvorlesung (im Folgenden kurz Vorlesung genannt) von zwei Sekretärinnen verwaltet werden. Auf das Sekretariat können außerdem alle Professoren zugreifen, die diese Vorlesung in einem beliebigen Semester gehalten haben bzw. halten. Die Vorlesung wird in jedem Semester von genau einem Professor gehalten, allerdings nicht immer von dem gleichen Professor. Jede Vorlesung wird durch mindestens ein Tutorium ergänzt, in das Studenten eingeteilt werden. Da in der Vorlesung keine Anwesenheitspflicht herrscht, verwaltet das System nur Studenten, die auch Tutorien besuchen. Die Tutorien werden von mindestens einem oder auch mehreren Assistenten betreut, die alle bei dem Professor angestellt sind, der die zugehörige Vorlesung hält.

Hat ein Student ein Tutorium erfolgreich absolviert, so erhält er einen Schein. Ein Student kann so oft an einem Tutorium teilnehmen, bis er es erfolgreich beendet und einen Schein erhält. Hat er einmal einen Schein erhalten, so kann er nicht mehr an einem weiteren Tutorium teilnehmen. Der Schein ist Voraussetzung für die Teilnahme an der Klausur, die auf dem Stoff des jeweiligen Semesters basiert.

In der Klausur werden Punkte vergeben, die zu Klausurnoten (im Folgenden KlNoten genannt) verarbeitet werden. Somit erhält jeder Student nach der erfolgreichen Teilnahme an einem Tutorium einen Schein und nach der Teilnahme an einer Klausur eine KlNote. Hat der Student die Klausur nicht erfolgreich bestanden, so ist die KlNote eine 5.0, und er muss die Klausur in einem folgenden Semester noch einmal schreiben.

Hat er die Klausur zum zweiten Mal nicht bestanden, so muss er die vom Professor angebotene mündliche Prüfung nutzen. Dazu redet der Student mit einer der beiden Sekretärinnen im Sekretariat. Die Sekretärin überprüft zunächst, ob der Student zur mündlichen Prüfung zugelassen ist.

Der Student ist zu einer mündlichen Prüfung zugelassen, wenn er schon zweimal die Klausur mitgeschrieben hat, aber nie eine bessere Note als 5.0 erreichte. Wenn dies der Fall ist, überprüft die Sekretärin, ob der gewünschte Professor einen Termin frei hat. Der gewünschte Professor sei derjenige, bei dem der Student die Vorlesung gehört hat. (Dabei ist zu beachten, dass alle Professoren am Semesterbeginn eine Anzahl von Terminen zu mündlichen Prüfungen festlegen, wobei diese Termine im System anfangs als nicht belegt gekennzeichnet sind.) Wenn der gewünschte Termin noch nicht belegt ist, soll der Termin gebucht und der Professor benachrichtigt werden. Er kann aber auch selber im Sekretariat nachsehen, welche Prüfungen demnächst anstehen.

Aufgabenstellung Entwickeln Sie die folgenden Teile des oben skizzierten Systems nach der FUSION-Methode und achten Sie dabei besonders auf die Systemoperationen $muendliche\_Pruefung\_beantragen$ und $termin\_buchen$.

1. Geben Sie die **Multiplizitäten** im gegebenen Klassenmodell an. *4 Punkte*

2. Vervollständigen Sie das vorgegebene **Systemklassenmodell**, indem Sie die Systemgrenze festlegen und gegebenenfalls Klassen und Assoziationen aus dem Klassenmodell übertragen oder neue Klassen und Assoziationen hinzufügen. Wenn neue Klassen oder Assoziationen hinzugefügt werden, so müssen diese auch benannt werden.

   6 Punkte

   Hinweis: Die Multiplizitäten sollen **nicht** in das Systemklassenmodell übertragen werden.

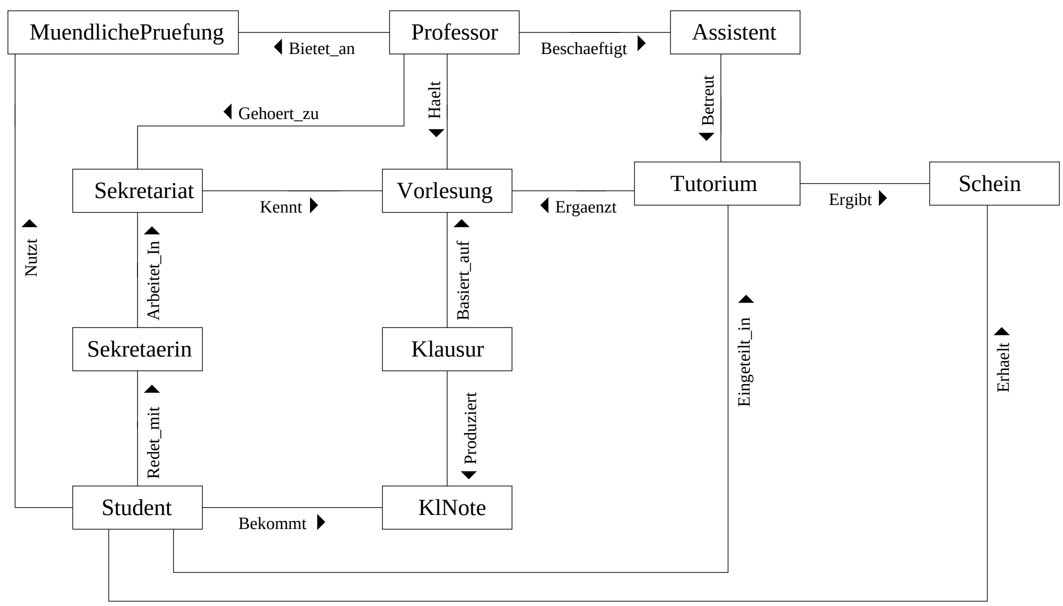
3. Geben Sie auf der Grundlage der Vorgaben den **Objektinteraktionsgraphen** für die Systemoperation $muendliche\_Pruefung\_beantragen$ an. *14 Punkte*

4. Vervollständigen Sie aufgrund der Vorgaben das **Operationsschema** für die Systemoperation $termin\_buchen$. *16 Punkte*

Hinweis: Stellen Sie sicher, dass sich Timeline-Diagramm, Operationsschemata und Objektinteraktionsgraph nicht widersprechen!

1 Klassenmodell

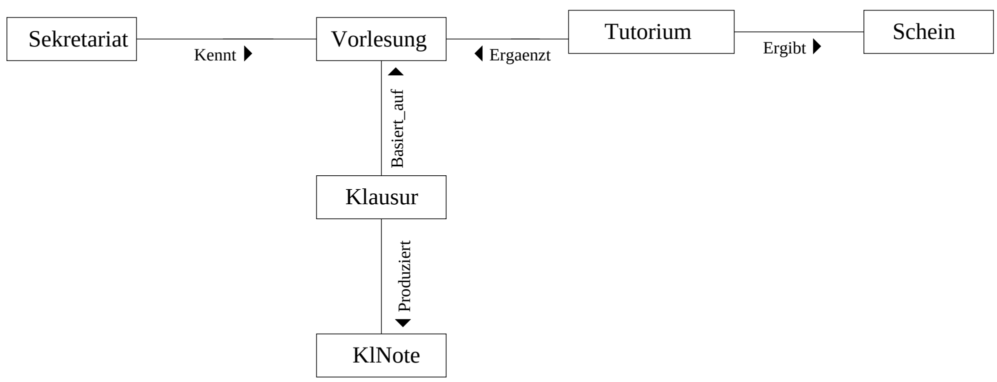
Geben Sie die *Multiplizitäten* im gegebenen Klassenmodell an.



2 Systemklassenmodell

Vervollständigen Sie das vorgegebene Systemklassenmodell, indem Sie die Systemgrenze festlegen und gegebenenfalls Klassen und Assoziationen aus dem Klassenmodell übertragen oder neue Klassen und Assoziationen hinzufügen. Wenn neue Klassen oder Assoziationen hinzugefügt werden, so müssen diese auch benannt werden.

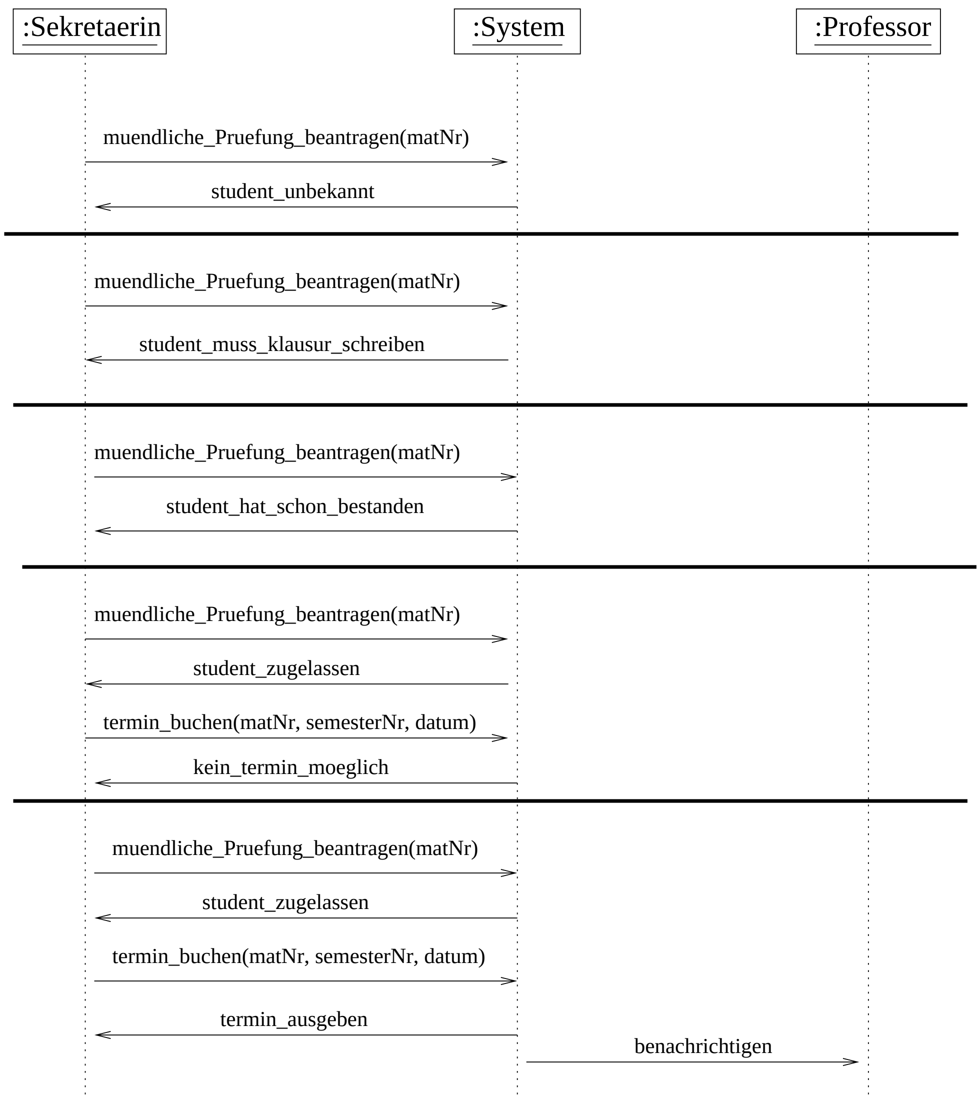
Hinweis: Die Multiplizitäten sollen **nicht** in das Systemklassenmodell übertragen werden.



3 Vorgaben

Sie benötigen diese Vorgaben zur Bearbeitung des Objektinteraktionsgraphen zur Systemoperation *muendliche_Pruefung_beantragen* und zur Erstellung des Operationsschemas der Systemoperation *termin_buchen*.
Hinweis: Es sollen nur die im Timeline-Diagramm vorgegebenen Fehlerfälle betrachtet werden.

3.1 Vorgaben: Timeline-Diagramm



3.2 Vorgaben: Attribute

Beachten Sie bitte auch, dass jeder Student eindeutig über seine Matrikelnummer identifiziert werden kann.

Vorlesung
semester : SEMESTER

KlNote
note : NOTE

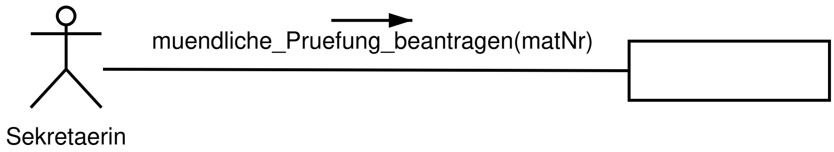
MuendlichePruefung
note : NOTE zeit : ZEIT datum : DATUM protokoll : TEXT belegt : BOOL

3.3 Vorgaben: Operationsschema

Name =	muendliche_Pruefung_beantragen
Desc =	Ein Student kann einen Termin für eine mündliche Prüfung beantragen. Wenn der Student mit der übergebenen Matrikelnummer dem System bekannt ist, bereits zwei Klausurnoten besitzt und beide Klausurnoten eine 5.0 sind, dann wird er zur mündlichen Prüfung zugelassen.
Input =	matNr: NAT
Reads =	sek:Sekretariat, Kennt_Student, Studentendaten, Bekommt, KlNote
Changes =	
Sends =	Sekretaerin:{student_unbekannt, student_muss_klausur_schreiben, student_hat_schon_bestanden, student_zugelassen}
Pre =	true
Post =	**let** bekannter_Student == {s:Studentendaten | s.matNr = matNr} • (bekannter_Student = ∅ ⇒ **is_sent**{student_unbekannt}) ∧ (bekannter_Student ≠ ∅ ⇒ **let** noten == {n:KlNote | ∃ s:bekannter_Student • (s,n) ∈ Bekommt} • (#noten ≤ 1 ⇒ **is_sent**{student_muss_klausur_schreiben}) ∧ (#noten = 2 ∧ (∃ n:noten • n.note < 5.0) ⇒ **is_sent**{student_hat_schon_bestanden}) ∧ (#noten = 2 ∧ (∀ n:noten • n.note = 5.0) ⇒ **is_sent**{student_zugelassen}))

4 Objektinteraktionsgraph

Stellen Sie auf der Grundlage des vorgegebenen Timeline-Diagramms und des vorgegebenen Operationsschemas der Systemoperation $muendliche\_Pruefung\_beantragen$ den dazugehörigen **Objektinteraktionsgraphen** auf.



5 Operationsschema

Vervollständigen Sie aufgrund des vorgegebenen Timeline-Diagramms und den Angaben zur Systemoperation $muendliche\_Pruefung\_beantragen$ das **Operationsschema** für die Systemoperation $termin\_buchen$.

Name =	termin_buchen
Desc =	Ein Student, der für eine mündliche Prüfung zugelassen ist, möchte einen mündlichen Prüfungstermin bei dem Professor buchen, bei dem er die Vorlesung gehört hat. Der Student gibt bei seiner Anfrage seine Matrikelnummer an, sowie das Semester, in dem er die Vorlesung gehört hat, und ein gewünschtes Datum. Da jeder Professor nur eine begrenzte Anzahl von mündlichen Prüfungsterminen anbietet, kann es sein, dass es zu dem übergebenen Datum keinen Prüfungstermin gibt oder bereits alle belegt sind. Wenn ein Prüfungstermin existiert, der zu diesem Datum nicht belegt ist, so wird dieser gebucht, der Sekretärin bestätigt und dem Professor mitgeteilt.
Input =	matNr: NAT, semesterNr:SEMESTER, datum:DATUM
Reads =	
Changes =	
Sends =	
Pre =	

Post =

Klausur SWT 25.2.2004, Name:__________________________ MatrNr.: __________

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel

*Klausur SWT 25.2.2004, Name:*__________________________ *MatrNr.:* __________

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel

Aufgabe 2 (Object-Z) - Skilift

Skifahrer können durch Benutzung eines Skilifts vom Tal auf den Berg fahren und vom Berg ins Tal. An dem Skilift hängen Gondeln, die jeweils einen Skifahrer transportieren. Das Einsteigen und das Aussteigen aus Gondeln findet sowohl im Tal als auch auf dem Berg in einem Schrankenbereich statt.

(Geschätzte Bearbeitungszeit: 40 min.)

Aufgabenstellung: Spezifizieren Sie den Skilift in Object-Z, indem Sie die vorgegebenen Klassendefinitionen ergänzen.

Folgende globale Definitionen sind vorgegeben:

$$[SKIFAHRER]$$
$$Nachricht ::= raus\_ok \mid raus\_nicht\_ok \mid rein\_ok \mid rein\_nicht\_ok \mid skifahrer\_ausgestiegen \mid kein\_skifahrer\_ausgestiegen$$

Gondel

$\boxed{\textit{10 Punkte}}$

Eine Gondel befördert einen Skifahrer auf dem Skilift. Sie ist durch eine Nummer gekennzeichnet, die zwischen 1 und 100 liegt. Wenn eine Gondel einen Skifahrer transportiert, dann ist sie belegt. Initial ist die Gondel nicht belegt. Die Operation *Belegen* modelliert das Einsteigen eines Skifahrers in eine leere Gondel. Die robuste und für die Umgebung sichtbare Operation *Belegung_weg* soll das Aussteigen eines Skifahrers aus einer Gondel spezifizieren. (Der Skifahrer muss nach einer Fahrt in einer Gondel vom Berg ins Tal oder umgekehrt immer aus der Gondel aussteigen. Dies soll durch den Wert der Zustandsvariablen *belegt* modelliert werden.) Spezifizieren Sie zusätzlich die Schnittstelle der Klasse.

Gondel

$\upharpoonright$ ( )

$nummer : \mathbb{N}$

INIT

Belegen

$\Delta(belegt, fahrer)$

$s? : SKIFAHRER$

Belegung_weg_ok

$\Delta(belegt)$

$s\_raus! : SKIFAHRER$

$nb! : Nachricht$

Belegung_weg_kein_Skifahrer

$Belegung\_weg \mathrel{\widehat{=}}$

Schrankenbereich

14 Punkte

Im Tal und auf dem Berg sind insgesamt zwei Schrankenbereiche vorhanden, wobei jeder jeweils aus einem Eingangs- und einem Ausgangsbereich besteht. Im Schrankenbereich befindet sich immer genau eine Gondel. Aus dieser muss der in der Gondel transportierte Skifahrer im Ausgangsbereich aussteigen, dann wird die leere Gondel in den Eingangsbereich weitergeschoben, und dort steigt ein wartender Skifahrer in die Gondel ein, oder die Gondel bleibt unbelegt. Der Eingangsbereich fasst maximal 50 Skifahrer und kann nur betreten werden, wenn die Eingangsdrehtür geöffnet ist. Wenn ein Skifahrer den Eingangsbereich betritt, reiht er sich in die Schlange der wartenden Skifahrer ein. So soll sichergestellt werden, dass der, der am längsten wartet, als erster in eine Gondel steigen kann.

Initial befinden sich keine Skifahrer im Eingangsbereich und die Eingangsdrehtür ist offen.

Die robuste Operation *Skifahrer_rein* modelliert den Eintritt eines Skifahrers in den Eingangsbereich abhängig vom verfügbaren Platz. Wenn der Eingangsbereich voll ist, dann wird die Eingangsdrehtür geschlossen.

Die Operationen *Skifahrer_raus_ok* und *Skifahrer_raus_nicht_ok* modellieren den Austritt eines Skifahrers aus dem Eingangsbereich (und damit aus der Schlange der wartenden Skifahrer), wenn kein anderer Skifahrer mehr vor ihm steht. Die beiden Operationen werden innerhalb der Operation *Gondel_neu_belegen* verwendet, die das Entfernen der Belegung der Gondel des Schrankenbereichs und die neue Belegung dieser Gondel mit dem am längsten im Eingangsbereich wartenden Skifahrer modellieren soll.

Die Operation *Gondel_setzen* vermerkt die übergebene Gondel im Schrankenbereich. Es soll sichergestellt werden, dass diese ungleich der Gondel ist, die sich vorher im Schrankenbereich befunden hat.

Hinweis: Sie können die Funktionen **min** und **max** verwenden, die das minimale bzw. das maximale Element einer Menge von natürlichen Zahlen ergibt.

Schrankenbereich

$maximal : \mathbb{N}$

$skifahrer : \mathbb{N} \rightarrowtail\!\!\!\!\!\!\rightarrow SKIFAHRER$

$gondel : Gondel$

$drehtür\_rein\_auf : \mathbb{B}$

INIT

Skifahrer_rein_ok

$\Delta(skifahrer, drehtür\_rein\_auf)$

$s\_rein? : SKIFAHRER$

$n\_rein! : Nachricht$

Skifahrer_rein_nicht_ok

$Skifahrer\_rein \mathrel{\widehat=}$

$Skifahrer\_raus\_ok$

$\Delta(skifahrer, drehtür\_rein\_auf)$
$s! : SKIFAHRER$
$n\_raus! : Nachricht$

$Skifahrer\_raus\_nicht\_ok$

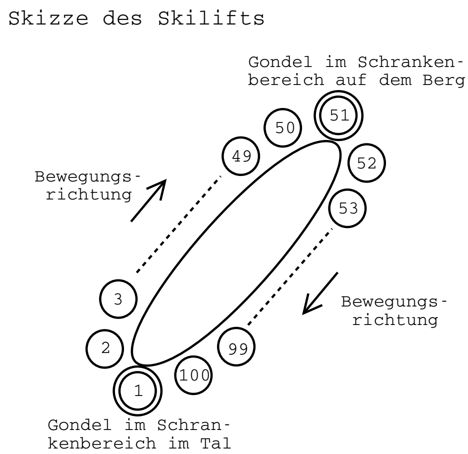
$Gondel\_neu\_belegen \mathrel{\widehat=}$

$Gondel\_setzen$

Skilift

11 Punkte

An einem Skilift hängen 100 Gondeln, die Skifahrer vom Tal auf den Berg befördern und vom Berg ins Tal. Auf dem Berg und im Tal befindet sich jeweils ein Schrankenbereich, um den Zutritt von Skifahrern zu den Gondeln des Skilifts zu kontrollieren. Die Gondeln des Skilifts sind von 1 bis 100 durchnummeriert und in aufsteigender Nummerierung aufgehängt (siehe Bild), und sie werden im Uhrzeigersinn bewegt.



Spezifizieren Sie jeweils eine invariante Eigenschaft hinsichtlich:

- der Anzahl der Gondeln,
- der Gondelnummern,
- der Schrankenbereiche,

und zwei invariante Eigenschaften hinsichtlich der Gondeln in den Schrankenbereichen.

Initial ist der Skilift nicht in Bewegung, und alle Gondeln des Skilifts und die beiden Schrankenbereiche befinden sich in ihrer initialen Konfiguration.

Die Operation *Anhalten* modelliert, dass der Skilift angehalten wird, was immer möglich sein soll. Die Operation *Weiterfahren* setzt den stillstehenden Skilift in Bewegung.

Die Operation *Neue_Gondeln_bestimmen* beschreibt die Auswahl der Gondeln, die die beiden Schrankenbereiche als nächste betreten.

Die Operation *Gondeln_aktualisieren* modelliert das gleichzeitige Vermerken von Gondeln in den beiden Schrankenbereichen mit Gondeln, die durch die Operation *Neue_Gondeln_bestimmen* bestimmt wurden.

Die bereits ausspezifizierte Operation *Gondeln_neu_belegen* modelliert die Neubelegung der Gondeln, die sich in den Schrankenbereichen befinden.

Die Operation *Gondeln_abarbeiten* modelliert den Gesamtvorgang des Abarbeitens von den sich in den Schrankenbereichen befindenden Gondeln. Gleichzeitig zum Anhalten des Skilifts sollen die Gondeln in den beiden Schrankenbereichen aktualisiert werden, danach sollen die beiden Gondeln neu belegt werden. Zum Schluss soll der Skilift wieder in Bewegung gesetzt werden.

Skilift

$gondeln : \mathbb{F}\ Gondel$
$in\_bewegung : \mathbb{B}$
$s\_tal : Schrankenbereich$
$s\_berg : Schrankenbereich$

INIT

Anhalten

Weiterfahren

$Neue\_Gondeln\_bestimmen$

$g\_tal! : gondeln$

$g\_berg! : gondeln$

$Gondeln\_aktualisieren \mathrel{\widehat=}$

$Gondeln\_neu\_belegen \mathrel{\widehat=}$

$\quad s\_tal.Gondel\_neu\_belegen \setminus (s\_raus!, n\_raus!, nb!) \land$

$\quad s\_berg.Gondel\_neu\_belegen \setminus (s\_raus!, n\_raus!, nb!)$

$Gondeln\_abarbeiten \mathrel{\widehat=}$

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel

*Klausur SWT 25.2.2004, Name:*__________________________ *MatrNr.:* __________

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel

Aufgabe 3 (Statecharts) - Waschmaschine

Eine Waschmaschine für Kleidung soll mit Hilfe von Statecharts modelliert werden. Sie besteht vereinfacht aus folgenden Komponenten:

- einer *Trommel*, in der die Wäsche Platz findet,
- einer *Tür*, durch die Wäsche eingefüllt und entnommen wird,
- einer *Türverriegelung*, die das Öffnen der Tür verhindern kann,
- einer *Wasserzufuhr* und
- einer *Steuerung*, die den Waschvorgang steuert.

Ein Benutzer der Waschmaschine kann diese über einen Knopf ein- und ausschalten und die Tür auf- und zumachen. Es gibt nur ein vorgegebenes Waschprogramm. Gestartet wird es durch Drücken des Ein/Aus-Knopfs im ausgeschalteten Zustand, wobei die Tür geschlossen sein muss. Daraufhin verriegelt die Maschine die Tür und startet das Waschprogramm.
Das Waschprogramm funktioniert wie folgt:

1) Füllen der Trommel:
Zunächst wird die Trommel mit Wasser gefüllt, indem die Wasserzufuhr für 4 Sekunden geöffnet und dann wieder geschlossen wird. Danach wird die Trommel 5 Sekunden nach rechts gedreht und wieder gestoppt. Diese Abfolge wird solange wiederholt, bis die Trommel 20 Liter Wasser enthält (Wasser-Drehen-Wasser-Drehen etc.). Bei geöffneter Wasserzufuhr fließt 1 Liter pro Sekunde in die Trommel. Modellieren sie die Wassermenge in der Trommel als Variable.

2) Waschen:
Ist die Trommel mit 20 Liter Wasser gefüllt, dreht sie sich 30 Minuten lang abwechselnd jeweils 1 Minute nach rechts und links (sie beginnt mit einer Rechtsdrehung). Die Dauer des Waschens (30 min) soll auch als Variable modelliert werden.

3) Abpumpen und Fertig:
Nach Ablauf der 30 Minuten wird die Trommel gestoppt, das Wasser abgepumpt (nur über die entsprechende Variable zu modellieren) und das Programm ist beendet. Nach weiteren 3 Minuten wird die Türverriegelung aufgehoben und die Tür lässt sich wieder öffnen. Bevor ein neuer Waschvorgang gestartet werden kann, muss die Maschine über den Ein/Aus-Knopf ausgeschaltet werden.

Wird während des Waschvorgangs der Ein/Aus-Knopf betätigt, unterbricht die Maschine das Waschprogramm. Eine Unterbrechung ist nur im Abschnitt "Waschen" des Waschprogramms möglich. Die Tür bleibt bei einer solchen Unterbrechung aus Sicherheitsgründen verriegelt. Bei erneutem Einschalten setzt die Maschine den Waschvorgang mit dem zuletzt aktiven Programmpunkt fort.
Im Initialzustand ist die Waschmaschine ausgeschaltet, die Tür geschlossen, die Türverriegelung nicht aktiv, die Trommel im Stillstand und die Wasserzufuhr gesperrt.

Aufgabenstellung

Modellieren sie die angegebenen Komponenten der Waschmaschine und deren Steuerung mit Hilfe von Statecharts.
Dokumentieren Sie alle verwendeten Ereignisse *kurz* in einer Tabelle.

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel

*Klausur SWT 25.2.2004, Name:*__________________________ *MatrNr.:* __________

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel