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

Kammüller
Santen
Jucknath

Klausur

Softwaretechnik

Wintersemester 02/03, 20. Februar 2003

Prüfen Sie zunächst, ob dieses Exemplar vollständig ist (15 Blätter mit Aufgaben ohne Deckblatt und 3 zusätzliche Blätter). 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) - Auftragsverwaltung für den britischen Geheimdienst

(Geschätzte Bearbeitungszeit: 50 min.)

Nach dem letzten Einsatz des Agenten James Bond - Codename 007 - wurde deutlich, dass dieser ausgesprochen kostenintensiv war. Da die britische Regierung sparen muss, hat sie nun ein für alle Doppel-Null-Agenten verbindliches Computersystem angefordert, in dem alle Aufträge der Doppel-Null-Agenten verwaltet werden sollen. Die anderen Agenten werden nicht mitverwaltet.

Für die 9 Doppel-Null-Agenten können folgende Personen einen Auftrag eingeben: Der Leiter des britischen Geheimdienstes (Codename M) und der britische Premierminister. M hat darüber hinaus den alleinigen Zugang zu den geheimen Unterlagen, die der Agent benötigt, um seinen Auftrag durchzuführen. Diese beinhalten die Auftragsbeschreibung und die Namen möglicher Kontaktpersonen. Jeder Agent kann nur an einem Auftrag gleichzeitig arbeiten, aber mehrere Agenten können an einem Auftrag zusammenarbeiten. Jeder neue Auftrag muss mit einem finanziellen Budget ausgestattet werden, was im Sekretariat von Miss Moneypenny erledigt wird. Sie ist auch diejenige, die einen Agenten mit Papieren (Pass & Visa) versorgt. Darüber hinaus kann der Agent - bei Bedarf - vom Leiter der Entwicklungsabteilung (Codename Q) Fahrzeuge und anderes Arbeitsmaterial erhalten.

Aufgabenstellung

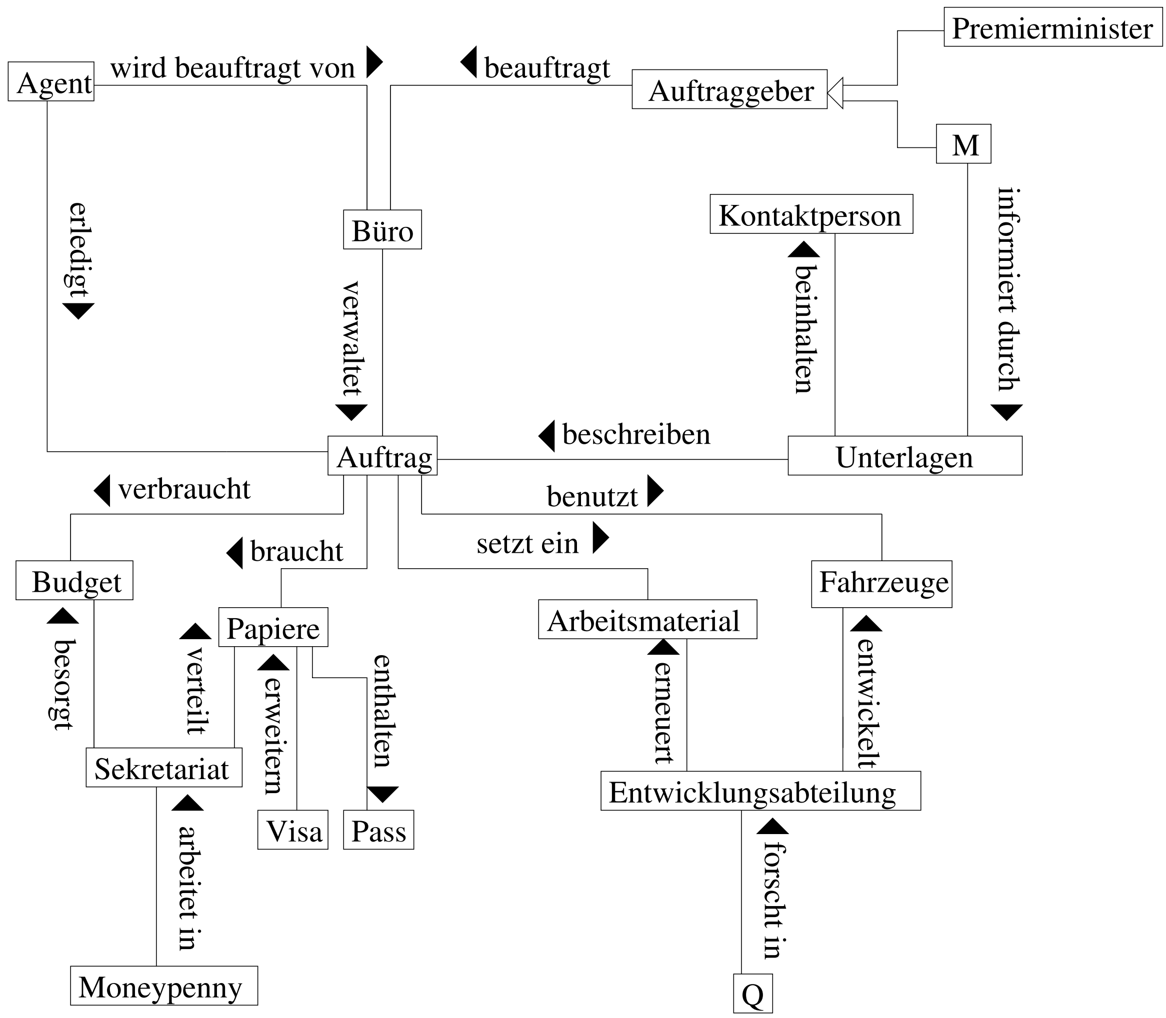
Die Entwicklung des Systems soll mit der Fusion-Methode durchgeführt werden. Dabei soll das Verhalten der Systemoperation *auftragseingabe* genau untersucht werden.

1. Geben Sie die *Multiplizitäten* im gegebenen Klassenmodell an. *2 Punkte*
2. Vervollständigen Sie das gegebene Use-Case-Modell, indem Sie die Akteure des Systems festlegen und ihnen Funktionsgruppen zuordnen. *6 Punkte*
3. Vervollständigen Sie das vorgegebene Systemklassenmodell, indem Sie die Systemgrenze festlegen und gegebenenfalls weitere neue Klassen und Assoziationen hinzufügen.
   Hinweis: Die Multiplizitäten sollen in das Systemklassenmodell **nicht** übertragen werden. *5 Punkte*

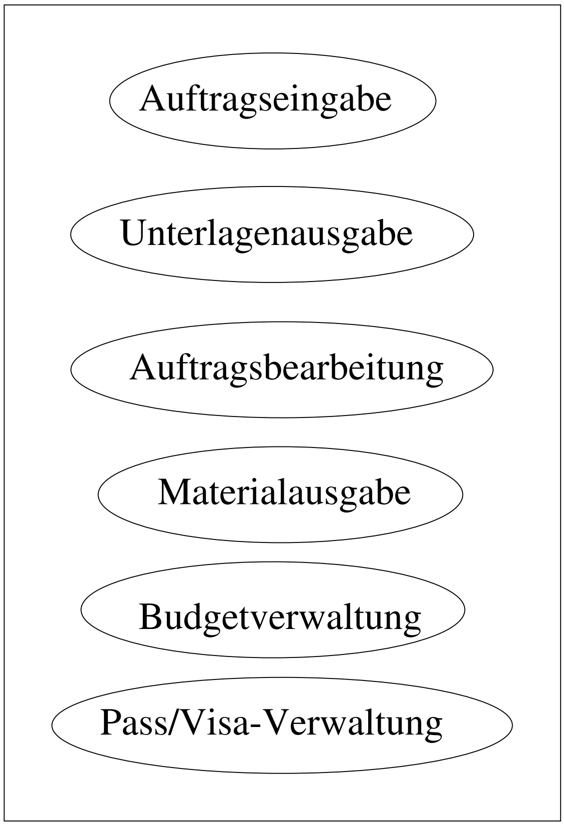
4. Vervollständigen Sie das *Operationsschema* für die Systemoperation $auftragseingabe$. 15 Punkte

5. Stellen Sie den Objektinteraktionsgraphen für die Systemoperation $auftragseingabe$ auf. 12 Punkte

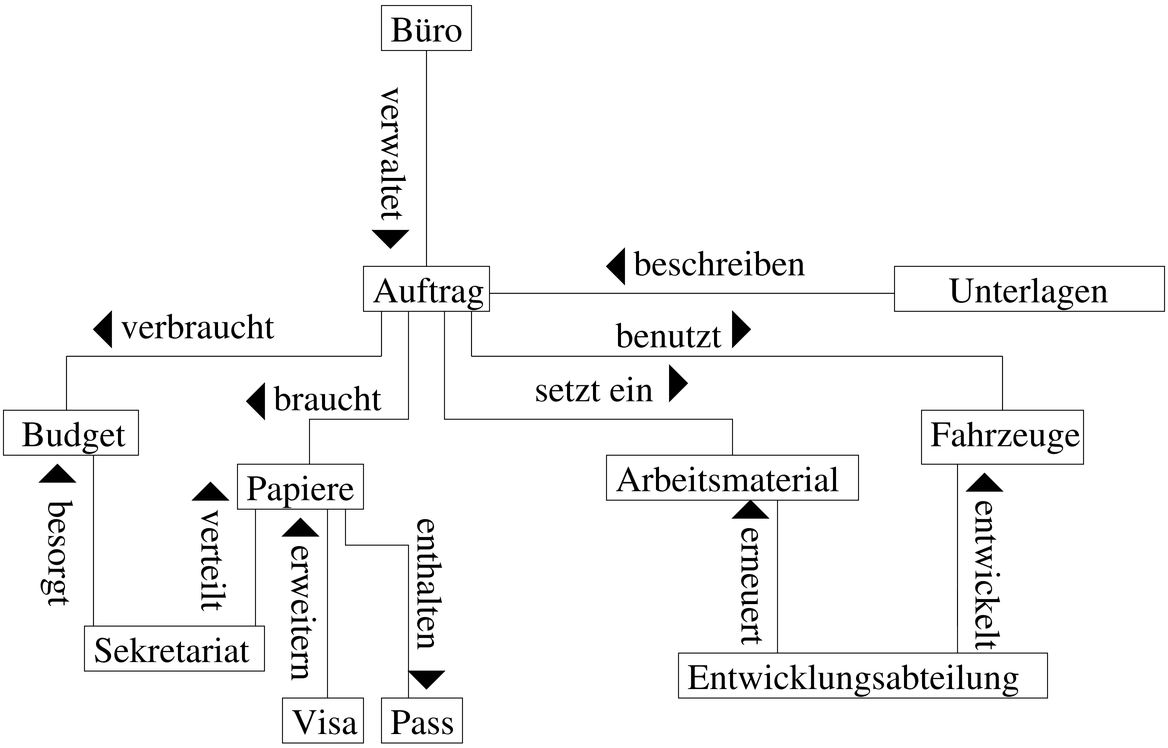
Klassenmodell



Use-Case-Modell



Systemklassenmodell



Operationsschema

Beim Spezifizieren der Operation $auftragseingabe$ sollen Sie die im folgenden aufgeführten Fälle berücksichtigen.

1. Wenn das für den Auftrag gewünschte Budget größer als die dem Sekretariat zur Verfügung stehenden Geldreserven ist, soll der Auftrag mit der Meldung **Auftrag zu teuer** abgelehnt und Miss Moneypenny die entsprechende Meldung **Geldreserven überprüfen** geschickt werden.
2. Wenn für den Auftrag mehr Agenten benötigt werden, als Agenten ohne Auftrag zur Verfügung stehen, soll der Auftrag mit der Meldung **Keine Agenten** an den Auftraggeber abgelehnt werden.
3. Andernfalls wird der Auftrag angenommen. Dann wird insbesondere ein Objekt zur Verwaltung der Unterlagen erzeugt und M aufgefordert die Unterlagen zu ergänzen (Meldung **Unterlagen ergänzen**).

Folgende **Attribute** sollen in der Spezifikation des Operationsschemas verwendet werden.

Sekretariat
geldreserven : NAT

Budget
kosten : NAT

Op. = *auftragseingabe*

Desc. = *Der Auftraggeber übergibt dem Büro einen neuen Auftrag mit den Angaben wieviele 00-Agenten eingesetzt werden sollen (agentenzahl) und wie groß die Kosten sind (kosten). Das Büro überprüft daraufhin, ob dieser Auftrag möglich ist und erzeugt ihn gegebenenfalls. Wenn der Auftrag nicht erstellt werden kann, werden entsprechende Fehlermeldungen ausgegeben.*

Input = *agentenzahl:NAT, kosten:NAT*

Reads =

Changes =

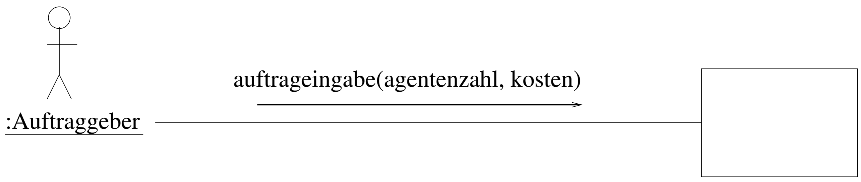
Sends =

Pre =

Post =

Objektinteraktionsgraph

Entwickeln Sie nun auf Basis des Operationsschemas der vorangegangenen Aufgabe den Objektinteraktionsgraph für die Systemoperation *auftragseingabe*.



Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel

Aufgabe 2 (Object-Z) - Postamt

(Geschätzte Bearbeitungszeit: 40 min.)

In dem Postamt gibt es eine Menge von Schaltern, an denen Beamte sitzen, die Kunden bedienen. Damit die Kunden nicht in einer Schlange anstehen müssen, gibt es einen Ticketautomaten, der Wartenummern ausgibt, wenn ein Kunde in das Amt eintritt (siehe unten). Für den reibungslosen Ablauf werden Kunden in der Reihenfolge der Wartenummern aufgerufen und bedient. In dem Postamt können nur Briefmarken (Porto) gekauft werden. Von den Werten der Marken bzw. ihrem Preis soll abstrahiert werden.

Aufgabenstellung

Spezifizieren Sie das Postamt in Object-Z, indem Sie die vorgegebenen Klassendefinitionen ergänzen.

Hinweis: Sie können die Funktion **max** verwenden, die das maximale Element einer Menge von natürlichen Zahlen ergibt.

Ticketautomat

$[Porto]$

```
┌─ Ticketautomat ─────────────────────────
│ ┌────────────────────────────────────
│ │ akt : ℕ
│ └────────────────────────────────────
│ ┌─ INIT ──────────────────────────────
│ │ akt = 0
│ └────────────────────────────────────
│ ┌─ ziehen ────────────────────────────
│ │ Δ(akt)
│ │ warte_nr! : ℕ
│ ├──────────
│ │ warte_nr! = akt
│ │ akt' = akt+1
│ └────────────────────────────────────
└──────────────────────────────────────
```

Kunde

Spezifizieren Sie die Klasse *Kunde*, indem Sie

- das *INIT*-Schema vervollständigen $\boxed{\textit{2 Punkte}}$
- und die Operation *porto_kaufen* definieren . $\boxed{\textit{3 Punkte}}$

Ein Kunde hat eine $geldbörse$, die anfangs nicht leer sein sollte und eine $tasche$, in die er gekaufte Marken steckt. Er hat weiterhin einen Schlangenplatz, der für die Koordination im Postamt benötigt wird.

Kunde

$geldbörse : \mathbb{N}$
$tasche : \mathbb{P}\, Porto$
$schlangenplatz : \mathbb{N}$

INIT

porto_kaufen

$\Delta(geldbörse, tasche)$
$geld! : \mathbb{N}$
$marke? : Porto$

Schalter

Spezifizieren Sie die Klasse *Schalter*. Vervollständigen Sie dazu

- das *INIT*-Schema, $\boxed{\text{2 Punkte}}$
- die Operationen *knopf_drücken* $\boxed{\text{2 Punkte}}$
- und *verkaufen*. $\boxed{\text{4 Punkte}}$

Die Operation *knopf_drücken* ermöglicht es den Schalter von frei auf besetzt, und umgekehrt, zu schalten. Beim Verkaufen soll eine Marke ausgegeben werden und etwas Geld eingenommen werden.

Schalter

$frei : \mathbb{B}$
$marken : \mathbb{P}\ Porto$
$kasse : \mathbb{N}$

INIT

knopf_drücken

$\Delta(frei)$

verkaufen

$\Delta(marken, kasse)$
$geld? : \mathbb{N}$
$marke! : Porto$

Postamt

Spezifizieren Sie nun die Klasse *Postamt*, indem Sie

- das Zustandsschema vervollständigen, *3 Punkte*
- das *INIT*-Schema vervollständigen, *3 Punkte*
- die Operation *einreihen* vervollständigen, *2 Punkte*
- die Operation *aufrufen* vervollständigen, *4 Punkte*
- die Operation *eintreten* definieren, *2 Punkte*
- die Operation *bedienen* definieren, *4 Punkte*
- und die Operation *abfertigen* definieren. *2 Punkte*

Die Operation *eintreten* beschreibt, dass ein Kunde hereinkommt, eine Wartenummer zieht und entsprechend in die Schlange aufgenommen wird. Die Operation *aufrufen* gibt einen freien Schalterbeamten und den nächsten wartenden Kunden aus. Kunden, die bedient werden, verlassen die Schlange und werden danach durch die Zustandsvariable *bedient* verwaltet. (Diese ist eine partielle, injektive Funktion). Die Operation *bedienen* umfasst das Verkaufen und Kaufen des Portos, sowie die entsprechenden Frei- bzw. Besetztmeldungen des betreffenden Schalters. Ein Arbeitsschritt im Ablauf des Amtes ist in der Operation *abfertigen* beschrieben, die aus *aufrufen* und *bedienen* besteht.

Frage: Welche Vorbedingung (informell) ist für einen neuen Kunden notwendig, damit das Postamt nicht blockiert wird? *2 Punkte*

Postamt

$beamte : \mathbb{F}\, Schalter$
$automat : Ticketautomat$
$schlange : \mathbb{F}\, Kunde$
$bedient : beamte \rightarrowtail\!\!\!\!\leftrightarrow Kunde$

INIT

einreihen

$\Delta(schlange)$
$k? : Kunde$
$warte\_nr? : \mathbb{N}$

aufrufen

$\Delta(bedient, beamte, schlange)$
$b! : beamte$
$k! : Kunde$

$eintreten \mathrel{\widehat=}$

$bedienen \mathrel{\widehat=}$

$abfertigen \mathrel{\widehat=}$

Bitte Verwendung dieses Blattes ankreuzen:

- [ ] Bearbeitung von Aufgabe _____
- [ ] Schmierzettel

Klausur SWT 21.2.2002, Name:__________________________ MatrNr.: __________

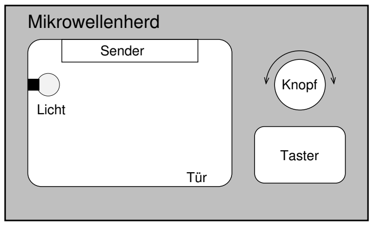
Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel

Aufgabe 3 (Statecharts) – Mikrowelle

(Geschätzte Bearbeitungszeit: 30 min.)



Betrachten Sie einen Mikrowellenherd mit folgenden Eigenschaften (siehe Bild): Die Tür des Herds wird über einen mechanischen Taster entriegelt. Durch kräftiges Drücken auf die Tür wird sie verriegelt. Der Herd besitzt einen Drehknopf ohne Anschläge, der sich rechts- und linksherum drehen lässt.

Wird die Tür geschlossen, kann innerhalb einer Minute durch Drehen des Knopfes nach rechts der Garvorgang mit einer Zeit von zunächst 15 Sekunden begonnen werden. Sobald der Knopf gedreht wird, beginnt der Garvorgang. Weiteres Drehen des Knopfes nach rechts erhöht die Garzeit in Schritten von 15 Sekunden. Die eingestellte Garzeit wird in einem Display oberhalb des Knopfes dargestellt. Durch Drehen des Knopfes nach links wird die Garzeit in Schritten von 15 Sekunden bis auf Null vermindert.

Wird die Tür geschlossen aber der Garvorgang nicht innerhalb einer Minute gestartet, dann kann mit dem Drehknopf keine Zeiteinstellung erfolgen. Um diese Sicherheitsfunktion aufzuheben, muss zunächst die Tür wieder geöffnet werden.

Während des Garvorgangs sind der Sender, der die Mikrowellen erzeugt, und das Licht im Garraum eingeschaltet. Die verbleibende Garzeit wird jeweils im Display angezeigt. Wenn die Garzeit abgelaufen ist, werden der Sender und das Licht im Garraum ausgeschaltet. Außerdem wird für 3 Sekunden ein Summton erzeugt.

Wird während der Garzeit die Tür geöffnet, werden Licht und Mikrowellensender ausgeschaltet. Die restliche Garzeit bleibt jedoch erhalten. In diesem Zustand kann durch Drehen des Knopfes die Garzeit verändert werden. Wird die Tür wieder geschlossen, wird der Garvorgang fortgesetzt.

Aufgabenstellung

Modellieren Sie den oben beschriebenen Mikrowellenherd mit den Komponenten Sender, Licht und Summer sowie die Steuerung dieser Komponenten mittels Statecharts. Verwenden Sie die Eingangsereignisse **to** und **ts** für das Öffnen und Schließen der Tür und die Eingangsereignisse **l** und **r** für das Links- und Rechtsdrehen des Knopfes. Verwenden Sie eine Zustandsvariable z, um die Garzeit zu speichern. *25 Punkte*

Das Display brauchen Sie **nicht** zu modellieren.

*Klausur SWT 21.2.2002, Name:*__________________________ *MatrNr.:* __________

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel

*Klausur SWT 21.2.2002, Name:*________________________ *MatrNr.:* __________

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe ____

☐ Schmierzettel