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

Jähnichen
Helke
Nordwig

Klausur

Softwaretechnik

24.7.2002

Prüfen Sie zunächst, ob dieses Exemplar vollständig ist (13 Blätter mit Aufgaben ohne Deckblatt und 3 zusätzliche Blätter). Es dürfen keine eigenen Blätter verwendet werden.

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

Aufgabe	1	2	3	Gesamt
Maximal:	40	40	20	100
Erreicht:				

Aufgabe 1 (Fusion) - Partyservice

(Geschätzte Bearbeitungszeit: 45 min.)

Ein expandierender Partyservice beauftragt eine Software-Firma mit der Entwicklung eines Verwaltungssystems, das die Abwicklung seiner Geschäftsprozesse unterstützen soll. Zusätzlich soll es für die Kunden möglich sein, per Handy über ein WAP-Portal mit dem System zu kommunizieren.

Bei dem Partyservice arbeiten mehrere *Mitarbeiter*, die die von den *Kunden* bestellten Produkte vorbereiten und ausliefern. Angeboten werden folgende Produkte: das *Gala-Dinner*, das *Große Buffet* und das *Kleine Buffet*. Jedes dieser Produkte benötigt verschiedene Arten und Mengen von *Vorräten*, die durch die Mitarbeiter beschafft, im System verwaltet und im *Lager* aufbewahrt werden müssen. Aufgrund der Lagerkapazität und der begrenzten Anzahl von Mitarbeitern können am gleichen Tag *maximal* zwei Produkte ausgeliefert werden.

Über das WAP-Portal sind genauere Informationen über die Produkte abrufbar. Interessiert sich ein Kunde für ein Produkt, so muss er sich zunächst über das WAP-Portal *registrieren*. Das System veranlasst dann die Überprüfung der Kundendaten durch den *Netzanbieter*. Ein Kunde ist genau einem Netzanbieter zugeordnet. Anschließend kann der Kunde eine verbindliche *Bestellung* aufgeben. Weichen seine Vorstellungen von den vorgegebenen Produkten ab, so kann er Extrawünsche während der Bestellung mitteilen. Die Bezahlung erfolgt via Handy-Rechnung, die der Netzanbieter dem Kunden zustellt.

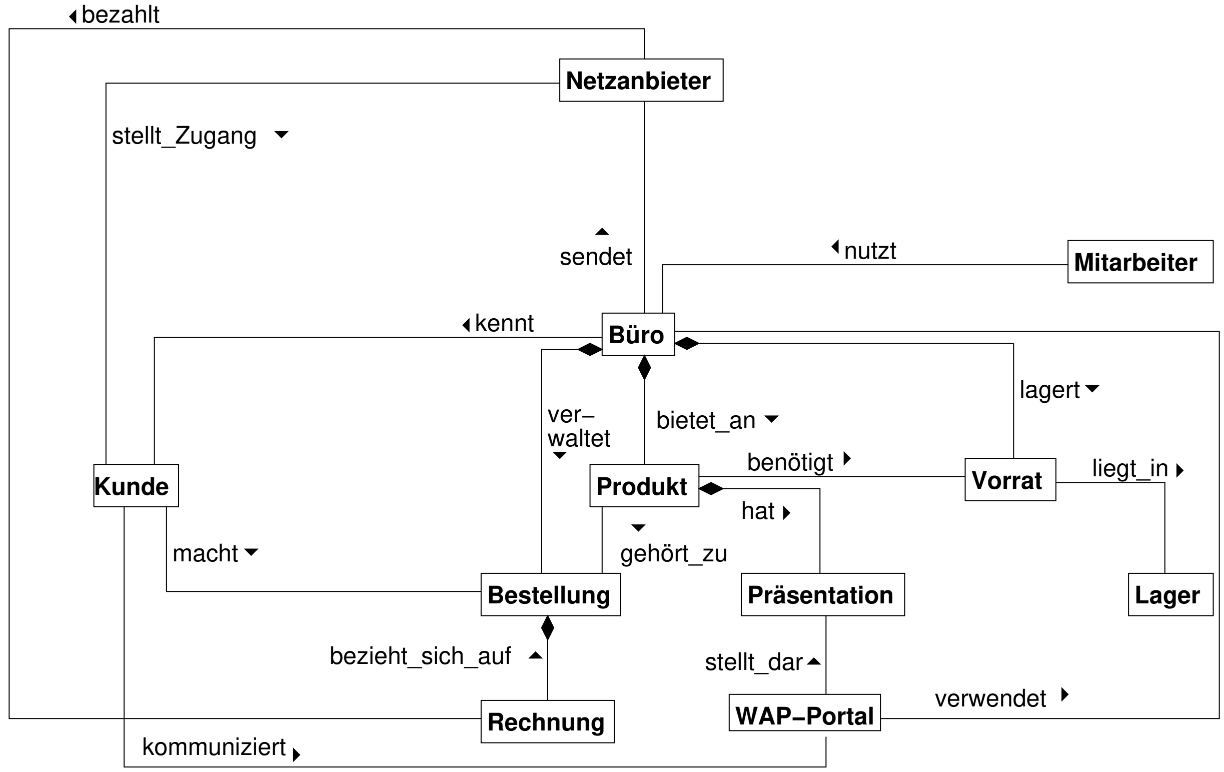
Wurde ein bestelltes Produkt ausgeliefert, so wird dies durch einen Mitarbeiter in der zugehörigen Bestellung vermerkt und eine *Rechnung* vom System an den Netzanbieter des Kunden via Email geschickt.

Aufgabenstellung

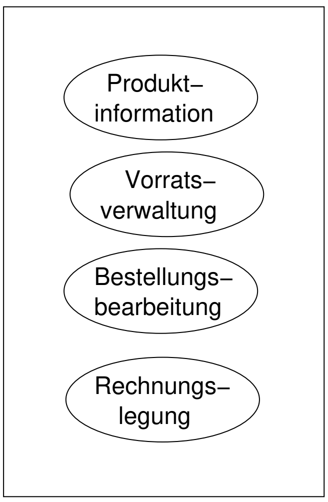
Das System soll nach der Fusion-Methode entwickelt werden. Dabei soll das Verhalten der Systemoperation *bestellen* genau untersucht werden.

- Ergänzen Sie im gegebenen Klassenmodell die Multiplizitäten. $\boxed{\textit{3 Punkte}}$
- Vervollständigen Sie das Use-Case-Modell, indem Sie die Akteure des Systems bestimmen und ihnen Funktionsgruppen zuordnen. $\boxed{\textit{4 Punkte}}$
- Vervollständigen Sie das vorgegebene Systemklassenmodell, indem Sie die Systemgrenze festlegen und gegebenenfalls weitere neue Klassen und zugehörige Assoziationen einfügen. $\boxed{\textit{8 Punkte}}$
- Vervollständigen Sie das Operationsschema für die Systemoperation *bestellen*. $\boxed{\textit{15 Punkte}}$
- Geben Sie den Objektinteraktionsgraphen für die Systemoperation *bestellen* an. Verwenden Sie das Konzept der Delegation. $\boxed{\textit{10 Punkte}}$

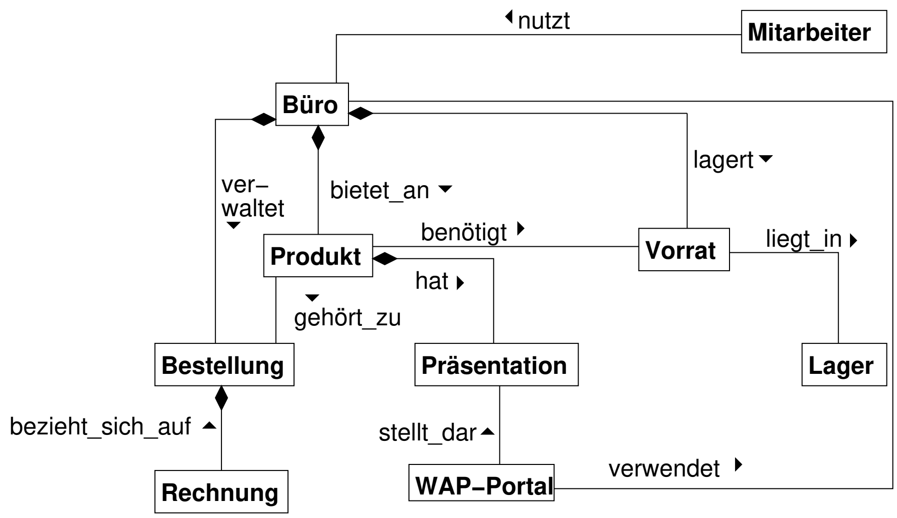
Klassenmodell



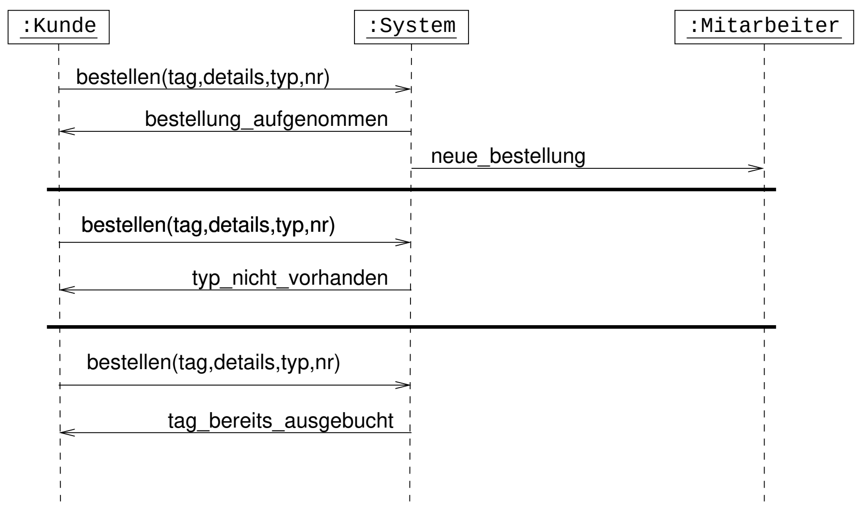
UseCase-Modell



Systemklassenmodell



TimeLineModell (Ausschnitt)



Operationsmodell (Ausschnitt)

Bei der Spezifikation der Systemoperation *bestellen* sollen Sie folgende Vorbedingungen annehmen:

- Das System kennt den Kunden mit Handynummer nr bereits, und somit auch seinen Netzanbieter.

Hinweis:

Nehmen Sie die folgenden Attribut- und Typdeklarationen an:

Produkt
typ:TYPE
pdat:PDATEN

Bestellung
status:STATUS
tag:TAG
details:DETAILS

KundenDaten
kdat:KDATEN
nr:HANDYNR

AnbieterDaten
aid:AID
adat:ADATEN

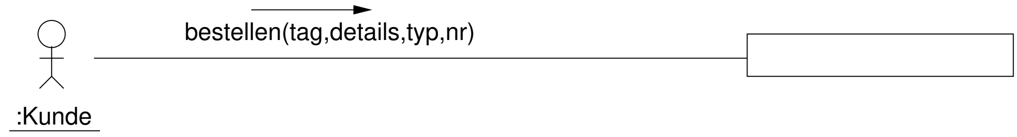
Elementare Datentypen:

TYP	Arten von Produkten
PDATEN	Beschreibungen der Produkte
TAG	Termin der Lieferung
DETAILS	Zeiten, Orte und Extrawünsche für Lieferungen
KDATEN	Ausführliche Kundendaten
HANDYNR	Handynummern
AID	IDs der Anbieter
ADATEN	Sonstige Daten der Anbieter
STATUS	Zustände der Bestellungen:

- *aktiv* - Bestellung aufgenommen
- *ausgeliefert* - Bestellung ausgeliefert
- *abgerechnet* - Rechnung verschickt
- *bezahlt* - Vorgang abgeschlossen

Op.	=	bestellen
Desc.	=	Ein Kunde bestellt ein Produkt. Falls ein solches Produkt nicht angeboten wird, oder der Partyservice bereits ausgebucht ist, wird eine entsprechende Fehlermeldung zurückgegeben.
Input	=	$tag : TAG;\ details : DETAILS;\ typ : TYP;\ nr : HANDYNR$
Reads	=	
Changes	=	
Sends	=	
Pre	=	
Post	=	

Objektinteraktionsmodell (Ausschnitt)



Aufgabe 2 (Object-Z) - Universal Serial Bus (USB)

(Geschätzte Bearbeitungszeit: 45 min.)

USB ist ein Bussystem, das es ermöglicht, verschiedene Arten von Peripheriegeräten (Devices) an einem PC zu betreiben. Ein wesentlicher Vorteil gegenüber anderen Bussystemen ist die Möglichkeit, zur Systemlaufzeit Geräte anschließen und entfernen zu können. Das Anschließen von Geräten wollen wir hier betrachten.

Um die Vielfalt dieser Geräte zu strukturieren, werden ähnliche Geräte, wie beispielsweise Eingabegeräte, jeweils einer Geräteklasse (*DevType*) zugeordnet. Für jede Geräteklasse gibt es einen zuständigen Treiber (*Driver*), der die Kommunikation mit den Geräten übernimmt, die ihm zugeordnet wurden. Der Betriebssystemkern (*Kernel*) verwaltet diese Treiber.

Aufgabenstellung

Spezifizieren Sie das beschriebene USB-System in Object-Z, indem Sie die vorgegebenen Klassendefinitionen ergänzen. Verwenden Sie dabei die gegebenen Typen:

$$[DevType]$$
$$State ::= plugged \mid unplugged$$
$$Message ::= devRegistered \mid devExists$$

Hinweis: Beachten Sie die Kommunikation zwischen Operationen der Klassen *Device*, *Driver* und *Kernel*.

Device

Peripheriegeräte werden durch die Klasse *Device* beschrieben. Sie können an ein System angeschlossen (*plugged*) oder noch nicht verbunden (*unplugged*) sein. Die Operation *Activate* modelliert das Anschließen eines Gerätes. Die Konstante *type* repräsentiert den Gerätetyp.

Spezifizieren Sie die Klasse *Device*, indem Sie die zugehörigen Schemata entwickeln.

7 Punkte

Device

$type : DevType$

INIT

Activate

$\Delta($ $)$

$device! : Device$

Driver

Jeder Treiber (*Driver*) bedient eine Menge von Geräten (*Devices*) der gleichen Geräteklasse. Wird ein Treiber-Objekt instanziiert, ist diese Menge zunächst leer.

Die Operation *Register* nimmt ein Gerät in diese Menge auf und gibt eine entsprechende Nachricht (*Message*) zurück. Es soll davon ausgegangen werden, dass die Operation Register nur mit Devices aufgerufen wird, deren Gerätetyp der Treiber unterstützt. Vervollständigen Sie die Spezifikation der Klasse *Driver* entsprechend dieser Vorgaben. 15 Punkte

Driver

$type : DevType$

INIT

RegisterOk

$\Delta(\qquad\qquad)$

$device? : Device$

DeviceExists

$device? : Device$

$Register \mathrel{\widehat{=}}$

Kernel

Der *Kernel* verwaltet alle aktiven Treiber. Da sämtliche Treiber zur Laufzeit geladen werden, sind nach Instanziierung dieser Klasse zunächst keine Treiber vorhanden. Für jeden Gerätetyp der aktiven Geräte gibt es genau einen Treiber, der diese Geräte behandelt.

Das Anschließen eines Gerätes *ConnectDevice* erfolgt in zwei aufeinanderfolgenden Schritten:

- Zunächst wird der zu einem Gerät gehörende Treiber geladen, falls dieser nicht schon geladen wurde (*Plug*).
- Anschließend wird das Gerät angeschlossen (aktiviert), und bei dem entsprechenden Treiber registiert (*Play*).

Vervollständigen Sie die Klasse *Kernel*, indem Sie Attribute, Invarianten und das Init-Schema angeben. $\boxed{\textit{5 Punkte}}$

Spezifizieren Sie unter Anwendung des Operationskalküls die Operationen *ConnectDevice*, *Plug*, *Play*, *LoadDriver* und *DriverExists*. $\boxed{\textit{13 Punkte}}$

Kernel

INIT

LoadDriver

$\Delta(\qquad\qquad)$

$device? : Device$

$driver! : Driver$

DriverExists

$device? : Device$

$driver! : Driver$

$Plug \mathrel{\widehat=}$

$Play \mathrel{\widehat=}$

$ConnectDevice \mathrel{\widehat=}$

Aufgabe 3 (Statecharts)

(Geschätzte Bearbeitungszeit: 25 min.)

Anrufbeantworter

Der Anrufbeantworter besitzt einen Nachrichtenspeicher, in dem aufgezeichnete Anrufe abgelegt werden können. Wenn der Speicher komplett belegt ist, werden keine weiteren Nachrichten angenommen.

Durch Betätigen der Taste $\boxed{>}$ können aufgezeichnete Nachrichten durch den Benutzer abgehört werden. Wenn keine Nachrichten aufgezeichnet wurden, ist diese Funktion deaktiviert. Nachrichten können weiterhin nur dann abgehört werden, wenn das Telefon nicht besetzt ist. Der Vorgang des Abspielens von Nachrichten wird sofort unterbrochen, wenn ein Anruf eingeht.

Das Betätigen der Taste $\boxed{C}$ bewirkt, dass alle gespeicherten Nachrichten gelöscht werden.

Wenn mindestens eine Nachrichten aufgezeichnet wurde, soll eine LED am Gerät blinken.

Der Anrufbeantworter schaltet sich nach zweimaligem Klingeln an, um eine Nachricht aufzunehmen. Der Aufnahmeprozess terminiert durch das Auflegen der anrufenden Person. Wird während der Aufnahme der Telefonhörer abgenommen, so wird der Aufnahmeprozess sofort gestoppt und der bis dahin aufgezeichnete Anruf abgespeichert.

Der Anrufbeantworter ist aus fünf Submodulen aufgebaut. Den zentralen Kern der Software bilden die Abspiel- und Anrufsteuerung. Diese beiden Steuereinheiten kommunizieren mit den drei anderen Subsystemen, die die Speicherverwaltung, die LED-Steuerung und den Telefonsensor beschreiben.

Aufgabenstellung

a) Modellieren Sie die Steuerungssoftware des Anrufbeantworters mit Hilfe von Statecharts. *15 Punkte*

b) Führen Sie in tabellarischer Form alle in ihrem Statechart-Modell verwendeten Ereignisse mit einer kurzen Erläuterung ihrer Funktion auf. *5 Punkte*

Verwenden Sie diese Tabelle zur Beschreibung der Ereignisse ihres Statecharts.

NAME	BESCHREIBUNG
AUSGABEEREIGNISSE	
EINGABEEREIGNISSE	
$\boxed{>}$	Betätigen der Abspieltaste
$\boxed{C}$	Betätigen der Löschtaste
INTERNE EREIGNISSE	

*Klausur SWT 24.7.2002, Name:*__________________________ *MatrNr.:* __________

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel

Klausur SWT 24.7.2002, Name:________________________ MatrNr.: __________

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe ____

☐ Schmierzettel

Bitte Verwendung dieses Blattes ankreuzen:

☐ Bearbeitung von Aufgabe _____

☐ Schmierzettel