Technische Universität Berlin
Institut für Softwaretechnik und Theoretische Informatik
FG Softwaretechnik
Ernst-Reuter-Platz 7
10587 Berlin

Jähnichen
Dobrev
Mehlhase
Rein

MPGI 3

Muster-Klausur A

Wintersemester 2009/2010
18. Februar 2010

Prüfen Sie zunächst, ob dieses Exemplar vollständig ist (*7* beidseitig bedruckte Blätter).

Tragen Sie auf diesem Titelblatt und darüber hinaus auf allen Blättern, die Sie für Ihre Niederschrift verwenden, Ihren Namen und Ihre Matrikelnummer ein. Zum Bestehen der Prüfungsleistung „Klausur“ müssen Sie in den beiden schriftlichen Leistungskontrollen insgesamt mindestens **50 Punkte** erreichen. Darüber hinaus müssen Sie in jeder der beiden schriftlichen Leistungskontrollen mindestens je 10 Punkte erreichen. Insgesamt sind in der vorliegenden schriftlichen Leistungskontrolle 50 Punkte möglich. Die Bearbeitungszeit beträgt 75 Minuten. Bevor die Bearbeitungszeit anfängt, haben Sie 15 Minuten Einlesezeit. In der Klausur sind außer eines beschriebenen DIN-A4-Blattes **keine Hilfsmittel** zugelassen. Verwenden Sie ausschließlich das ausgeteilte Klausur-Papier. Es dürfen nur **permanent-schwarze oder -blaue Stifte** zum Lösen der Aufgaben verwendet werden.

Viel Erfolg!

Name, Vorname: _____

Matrikelnummer: _____

Studienrichtung: _____

MPGI3-Übungen habe ich im _____ besucht.
(z.B. WS 2009/10)

Aufgabe	1	2	3	4	5	6	Gesamt
Maximal:	4	4	8	12	13	9	50
Erreicht:							

Aufgabenstellung

Entwickeln Sie das unten beschriebene System nach der in der Veranstaltung vermittelten Methodik. Ergänzen Sie hierzu die geforderten Artefakte in den nachfolgenden Aufgaben.

Die Stadttouristen

Die Stiftung „Stadttourismus“ möchte durch ein neuartiges Angebot junge Leute fördern, die mit wenig finanziellen Mitteln Städte zu touristischen Zwecken besuchen wollen.

Es soll ein Netzwerk aus *Gastgeber*n in ganz Europa aufgebaut werden, die kostenlose Unterkunft in den freien *Betten* ihrer *Wohnungen* anbieten. Die Gastgeber, die teilnehmen wollen, sollen ein Papierformular ausfüllen und an die Stiftung schicken. In diesem Formular beschreiben sie die Wohnungen, in denen sie freie Betten haben. Eine der Sekretärinnen der Stiftung überprüft an ihrem *Bürorechner* die Daten auf Vollständigkeit und Korrektheit und pflegt sie ins System ein. Es werden ausschließlich Gastgeber eingepflegt, die mindestens eine Wohnung mit je mindestens einem Bett anbieten.

Touristen, die einen oder mehrere Tage in einer Stadt verbringen möchten, müssen sich einmalig auf der *Webseite* `stadttourismus.de` mit ihren persönlichen Daten (Name, Email, Telefonnummer) registrieren. Nach der Registrierung bekommen sie Zugangsdaten (Benutzerkennung und Passwort), mit denen sie sich auf der Webseite einloggen können. Nur nach einem erfolgreichen Einloggen können sie die Liste der angebotenen Betten sehen. Es ist auch möglich, auf der Webseite Informationen über die Wohnungen zu bekommen. Entscheidet sich ein Tourist für ein Bett, muss er es für den gewünschten Zeitraum der geplanten Unterkunft reservieren. Ein Bett kann nur dann reserviert werden, wenn es für alle Tage des geplanten Aufenthalts frei ist. Ist dies der Fall, bekommt der Tourist eine Bestätigung. Anderenfalls wird er informiert, dass die Reservierung des Bettes für diesen Zeitraum nicht möglich ist.

Kommt eine Reservierung zustande, so wird eine der Sekretärinnen der Stiftung auf einem der *Bürorechner* informiert, dass eine neue Reservierung getätigt wurde. Sekretärinnen können jederzeit alle neuen Reservierungen anzeigen lassen und die betroffenen Gastgeber per Telefon über die bevorstehende Übernachtung informieren. Der Gastgeber kann dann die Reservierung telefonisch bestätigen oder ablehnen. Bei einer Ablehnung muss die Sekretärin die Reservierung löschen und den Touristen sofort per E-Mail informieren.

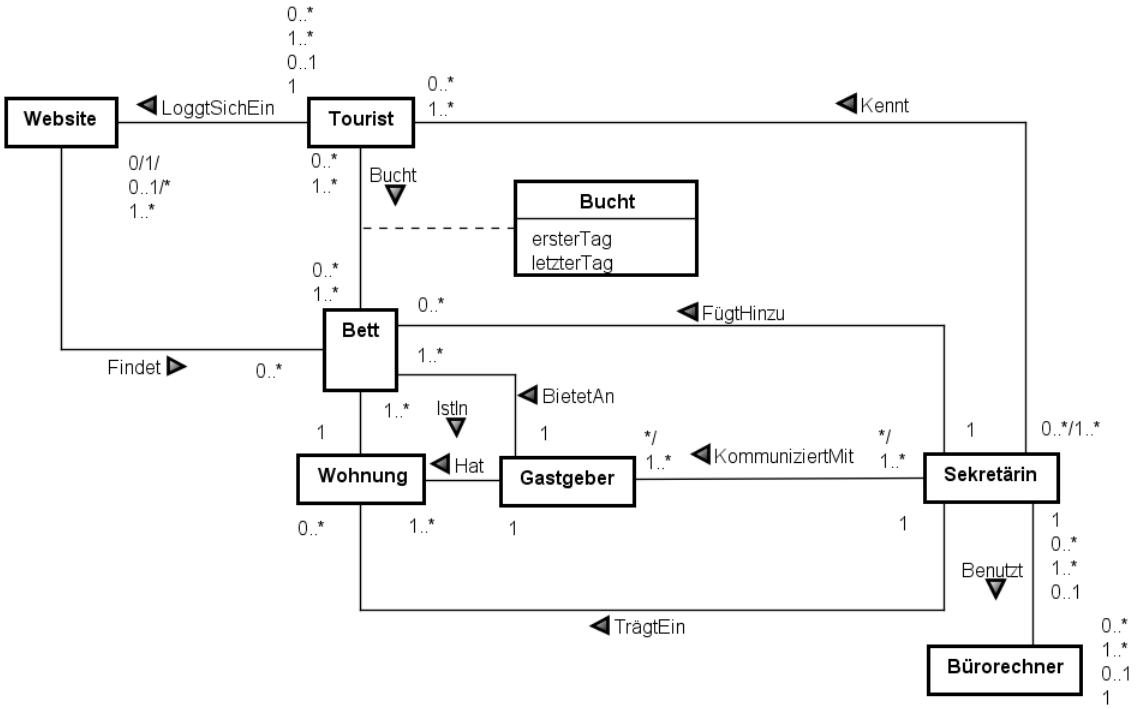
1. Klassenmodell des Gegenstandsbereichs

4 Punkte

Ergänzen Sie **alle** fehlenden Multiplizitäten im vorgegebenen Klassenmodell des Gegenstandsbereichs.

Die auf dem Diagramm gezeichneten Dreiecke (▶) kennzeichnen die Leserichtung der Assoziationsnamen.

Lösung:

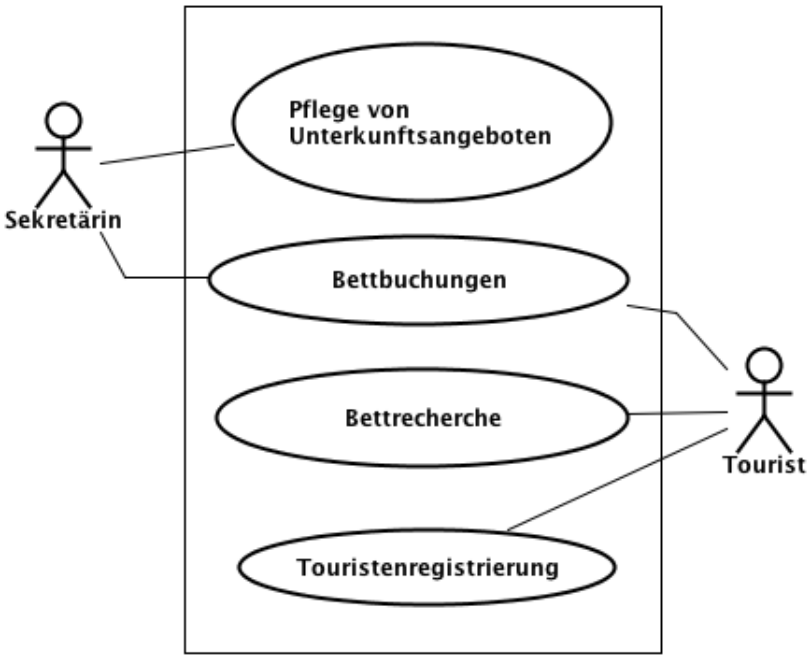


2. Use-Case-Diagramm *4 Punkte*

Vervollständigen Sie das vorgegebene Use-Case-Diagramm:

- Bestimmen Sie die Akteure des Systems und ordnen Sie diese den Funktionsgruppen zu.
- Fügen Sie der Vorgabe keine neuen Funktionsgruppen hinzu.
- Beziehungen zwischen den Funktionsgruppen dürfen hinzugefügt werden.

Beispiellösung:



3. Systemklassenmodell / Entwurfsklassenmodell

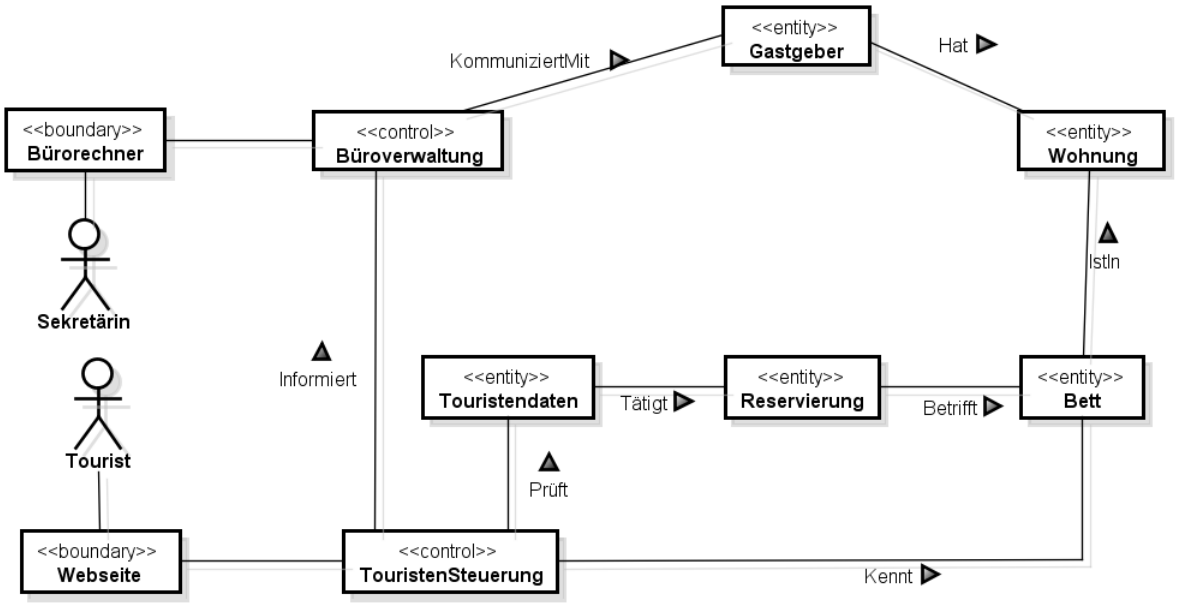
8 Punkte

Vervollständigen Sie das vorgegebene Systemklassenmodell auf der nächsten Seite:

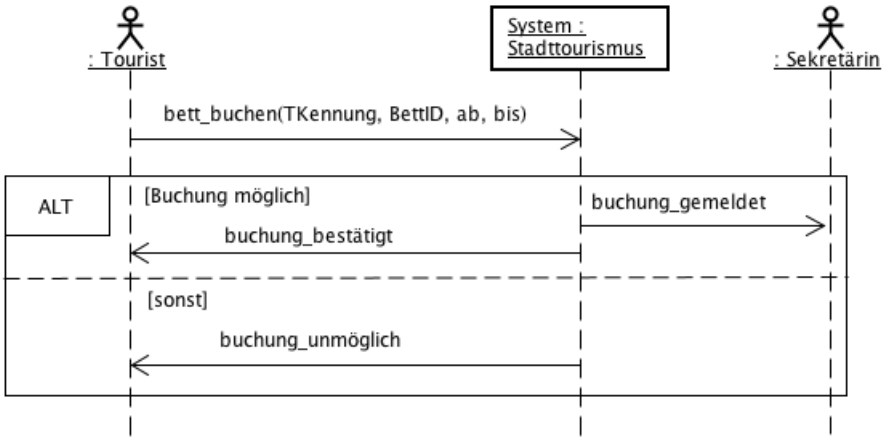
- Fügen Sie alle Akteure, Klassen und Assoziationen hinzu.
- Kennzeichnen Sie **alle** Klassen als *Boundary*, *Control* oder *Entity*.
- Benennen Sie alle Assoziationen.
- Verzichten Sie auf unnötige und redundante Assoziationen, um die Bearbeitung der nachfolgenden Artefakte zu erleichtern!
- Benutzen Sie keine Assoziationsklassen! *Hinweis:* Auf dem Diagramm wurde die **Assoziations**klasse *Bucht* bereits durch die normale Klasse *Reservierung* mit den Attributen *ersterTag* und *letzterTag* ersetzt.

Multiplizitäten und weitere Attribute müssen **nicht** eingetragen werden, können aber zum besseren Verständnis der nachfolgenden Artefakte beitragen.

Beispiellösung:



Sequenzdiagramm *bett_buchen* *keine Punkte*



Die Eingabeparameter der Systemoperation *bett_buchen* haben folgende Bedeutungen:

TKennung	die Benutzerkennung eines Touristen
BettID	die eindeutige Identifikationsnummer des Bettes
ab	der erste Tag der Unterkunft
bis	der letzte Tag der Unterkunft

Die Bedingung *Buchung möglich* drückt aus, dass es keine Buchungen für dieses Bett gibt, die sich zeitlich mit der gewünschten Buchung überschneiden.

4. Operationsschema *bett_buchen* *12 Punkte*

Vervollständigen Sie auf Basis Ihres Systemklassenmodells das vorgegebene Operationsschema für die Systemoperation *bett_buchen* auf der nächsten Seite. Es sollen nur die im Sequenzdiagramm dargestellten Fälle abgedeckt werden.

- Nehmen Sie die folgenden Attribut- und Typdeklarationen an:

Reservierung
ersterTag: Date letzterTag: Date ...

Bett
id: int ...

Touristendaten
kennung: String eingeloggt: boolean ...

- Benutzen Sie zum Vergleich von *Date*-Werten die mathematischenVergleichsoperationen ($\leq, <, >,$ und $\geq$).
- Beachten Sie die Vorbedingungen, die im **Desc.**-Teil beschrieben sind.

Op.	=	bett_buchen
Desc.	=	Ein Tourist bucht ein bestimmtes Bett für einen angegebenen Zeitraum. Es wird davon ausgegangen, dass der Tourist mit der Kennung *TKennung* im System bekannt ist und auf der Webseite korrekt eingeloggt ist. Die Existenz eines Bettes mit der Identifikation *BettID* wird vorausgesetzt.
Input	=	$TKennung : String, BettID : int, ab : Date, bis : Date$
Reads	=	$ts : TouristenSteuerung, Kennt, Bett, Touristendaten, Prüft,$ $b : Bett$ **with** $b.id = BettID \ \wedge (b, ts) \in Kennt$**,** $t : Touristendaten$ **with** $t.kennung = TKennung$ $\wedge\ (ts, t) \in Prüft \ \wedge t.eingeloggt$**,**
Changes	=	$Reservierung, Tätigt, Betrifft,$ $r : Reservierung$ **type**
Sends	=	Tourist: {buchung_bestätigt, buchung_unmöglich}, Sekretärin: {buchung_gemeldet}
Pre	=	*implicit*
Post	=	**let** $bettreservierungen == \{r : Reservierung \mid (r, b) \in Betrifft\}$ $überschneidende\_buchungen == \{r : bettreservierungen \mid$ $(ab \leq r.ersterTag \leq bis) \vee$ $(ab \leq r.letzterTag \leq bis) \vee (r.ersterTag \leq ab \wedge bis \leq r.letzterTag)\};$ $buchung\_möglich == \#überschneidende\_buchungen = 0 \wedge ab \leq bis;$ $buchung\_nicht\_möglich == \neg buchung\_möglich;$ • $(buchung\_nicht\_möglich \Rightarrow$ **is_sent** $\{buchung\_unmöglich\} \wedge$ **no_effect** $) \wedge$ $(buchung\_möglich \Rightarrow$ **is_sent** $\{buchung\_bestätigt\} \wedge$ **is_sent** $\{buchung\_gemeldet\} \wedge r$ **new** $\ \wedge$ $r.ersterTag' = ab \ \wedge r.letzterTag' = bis \wedge$ $Tätigt' = Tätigt \cup \{(t, r)\} \ \wedge Betrifft' = Betrifft \cup \{(r, b)\})$

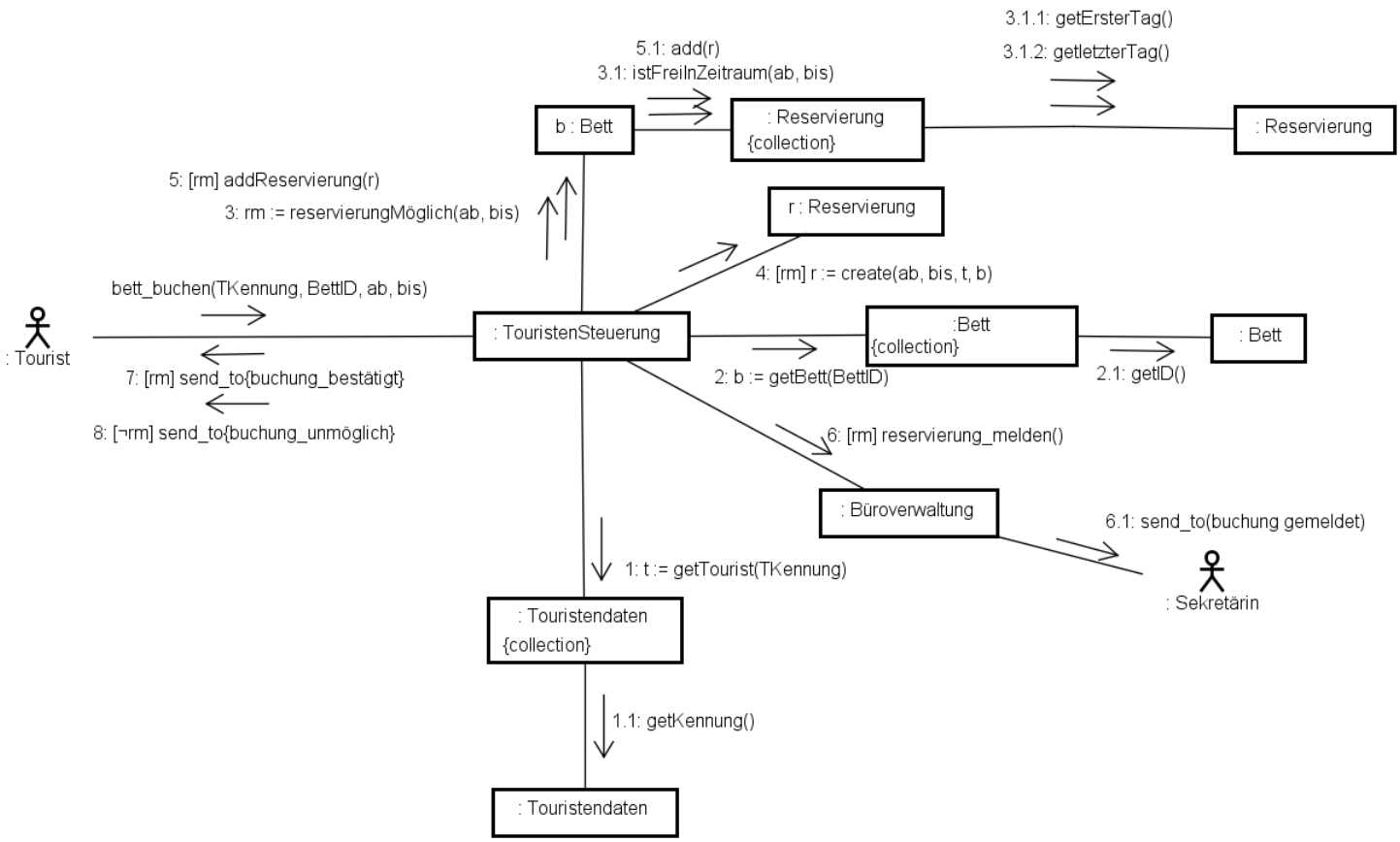
5. Kommunikationsdiagramm *bett_buchen* *13 Punkte*

Vervollständigen Sie auf Basis Ihres Systemklassenmodells und Ihres Operationsschemas das Kommunikationsdiagramm auf der nächsten Seite für die Systemoperation *bett_buchen*.

Beachten Sie dabei folgende Anweisungen:

- Für diese Systemoperation wird die Existenz eines Touristen mit der Kennung *TKennung* und die Existenz eines Bettes mit der Identifikation *BettID* vorausgesetzt.
- In der Systemoperation muss überprüft werden, ob für das gewünschte Bett bereits Reservierungen vorhanden sind, die sich mit dem gewünschten Zeitraum der Unterkunft überschneiden.
- Eine neu erstellte Reservierung muss dem Bett bekannt gemacht werden.

Beispiellösung:



6. Implementierungsklassenmodell

9 Punkte

Vervollständigen Sie den vorgegebenen Ausschnitt aus dem Implementierungsklassenmodell.

- Fügen Sie ausschließlich die Attribute und Methoden hinzu, die sich aus den vorangegangenen Artefakten für die Systemoperation *bett_buchen* ergeben.
- Benutzen Sie in den Controlklassen die vorgegebene Klasse *BettCollection*.
- Schränken Sie die Sichtbarkeiten der Klassenelemente auf das Mindestmaß für die Systemoperation *bett_buchen* ein.
- Collections-Eigenschaften wie *ordered* und *unique* müssen **nicht** angegeben werden!

Bett

- id: int
- reservierungen: Reservierung[*]
- istFreiInZeitraum(ab: Date, bis: Date) : Boolean

+ getID(): int
+ addReservierung(r: Reservierung): void
+ reservierungMöglich (ab: Date, bis: Date): boolean

BettCollection

- betten: Bett[*]

+ getBett(bettID: int): Bett

Touristendaten

- kennung: String
- eingeloggt: boolean

+ getKennung(): String

Reservierung

- ersterTag : Date
- letzterTag: Date
- t: Tourist

+ create (ab: Date, bis: Date, t: Touristendaten, b: Bett): Reservierung
+ getErsterTag(): Date
+ getLetzterTag(): Date

TouristenSteuerung

- touristen: TouristenDaten[*]
- betten: BettCollection
- bv: Büroverwaltung

+ bettBuchen(kennung: String, bettID: int, ab:Date, bis: Date): void
- getTourist(kennung: String): Tourist

Büroverwaltung

+ reservierung_melden():void