PA1415 Programvarudesign

Second Resit

Mikael Svahnberg (0455-385811)

2017-01-02

Poäng

(Fylls i av rättaren)

Fråga:	1	2	3	4	5	6	7	8	9	10	SUM
Maxpoäng:	3	3	3	3	3	6	4	4	4	4	37
Poäng:											

Betyg:	

Instruktioner

- **Vänligen kom ihåg** att ge ett svar i *varje* ruta [ ].
- Alla flervalsfrågor ger totalt 3 poäng, där varje rätt svar är värt 0.5p
- I en del frågor förväntas du rita ett diagram (till exempel ett klassdiagram eller ett interaktionsdiagram). Vänligen använd ett separat papper och gör en skiss först, och rita sedan in dem i den markerade rutan på tentan. Försök ordna elementen (och i synnerhet linjer mellan element) så att diagrammen blir lätta att läsa.
- **Tillåtna Materiel:** Penna, radergummi, linjal, glatt humör.

Lycka Till!

Fråga 1. Utvecklingsmetodologier 3P

Markera *varje* påstående nedan som antingen sant (**T** eller **S**) eller falskt (**F**):

- [F] Ett *Use Case* är UML-namnet på en testspecifikation.
- [F] *Vattenfallsmodellen* går ut på att man hela tiden går tillbaka till tidigare utvecklingsfaser och lägger till mer information.
- [T] Klasser och interaktion mellan objekt modelleras under *Design*-fasen.
- [F] Under *Analys*-fasen analyserar man sin design för att se till att den går att implementera.
- [T] I SCRUM samlar man alla sina krav i en prioriterad *backlog*.
- [F] Man skriver *testfall* baserat på vad man implementerat.

Fråga 2. Interaktionsdiagram 3P

Markera *varje* påstående nedan som antingen sant (**T** eller **S**) eller falskt (**F**):

- [F] Ett *sekvensdiagram* visar i vilken ordning olika interaktionsdiagram skall läsas.
- [T] Ett *systemsekvensdiagram* är bara ett specialfall av ett sekvensdiagram.
- [T] `garfield:Katt` är ett objekt av typen `Katt` som lagras i en variabel `garfield`.
- [F] Interaktionsdiagram visar vilka meddelanden som skickas mellan klasser.
- [T] Kommunikationsdiagram presenterar samma sak som ett sekvensdiagram.
- [F] I *sekvensdiagram* visar man det nuvarande värdet på alla attribut i alla inblandade objekt.

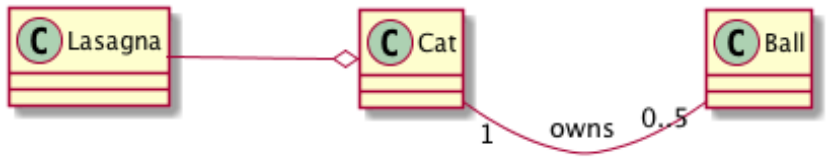
Fråga 3. Klassdiagram 3P

Markera *varje* påstående nedan som antingen sant (**T** eller **S**) eller falskt (**F**):

- [F] Ett klassdiagram beskriver det nuvarande värdet på alla attribut i alla inblandade klasser.
- [F] En association mellan två klasser kräver att båda klasserna känner till varandra.
- [F] Det får bara finnas ett enda objekt av typen `«Interface» Odie` i ett system.
- [T] Metoder i en klass kan vara `public`, `protected`, eller `private`.
- [F] En metod som är deklarerad som `public` får inte använda sig av attribut som är `private`.
- [F] `[Class A]`♦— `[Class B]` betyder att `B` ärver från `A`.

Fråga 4. Relationer mellan Klasser 3P

Studera diagrammet nedan, och markera sedan *varje* påstående nedanför som antingen sant (**T** eller **S**) eller falskt (**F**):



- [T] En `Cat` kan äga upp till fem objekt av typen `Ball`
- [T] En `Ball` har en och endast en ägare.
- [T] `blue:Ball` kan flyttas från `garfield:Cat` till `heatcliff:Cat`.
- [F] Associationen mellan `Cat` och `Ball` implementeras bäst som en array inuti `Ball`.
- [F] Det kan finnas högst 5 objekt av typen `Ball` i det här systemet.
- [F] Ett objekt av typen `Cat` kan inte existera utan objekt av typen `Lasagna`.

Fråga 5. GRASP-patterns 3P

Markera *varje* påstående nedan som antingen sant (**T** eller **S**) eller falskt (**F**):

- [F] En *Information Expert* är en klass som samlar all data i systemet som ett gränssnitt mot en databas.
- [T] En *Controller* tar emot systemhändelser och delegerar till andra klasser för att genomföra den begärda funktionen.
- [T] *High Cohesion* betyder att varje klass skall ha få och väldefinierade ansvarsområden.
- [F] En *Creator* är en klass som bara används när ett system startar upp första gången.
- [T] *Low Coupling* gör det enklare att förstå hur varje enskild klass fungerar.
- [F] En *Controller* är en klass som skall säkerställa att alla ekonomiska funktioner fungerar som de skall i ett program.

Fallbeskrivning för Återstående Frågor

Resten av frågorna använder sig av föjnade fall. Paragraferna är numrerade för att det skall vara lättare att referera till dem i frågorna. Läs beskrivningen noga innan du besvarar frågorna nedan.

Ticket Management System (TMS)

(1) *Tickets-R-Us* is an event management company. They sell tickets to *Football matches*, *Theater shows*, *Concerts*, as well as *Bus-* and *Train-* tickets to and from these events.

(2) Their business idea is to be a door-to-door supplier for these events. You can visit one of their offices, call them, use their mobile phone app, or their regular web app, and get the same service to book tickets for an event.

(3) If the customer already know what event they want to visit, they can order your tickets directly. In this case, they will be offered different date suggestions, as well as suggestions on how to get there and back. The customer will also get suggestions about e.g. nice restaurants or pubs nearby where you can meet and eat before or after the event. For a football match, pubs before the game should be suggested, For theater shows and concerts, restaurants after the show should be suggested.

(4) Customers can also contact Tickets-R-Us without knowing exactly what they want to do. In this case, the customer will be offered a list of events on a particular date, or a list of events based on categories or keywords that the customer provides.

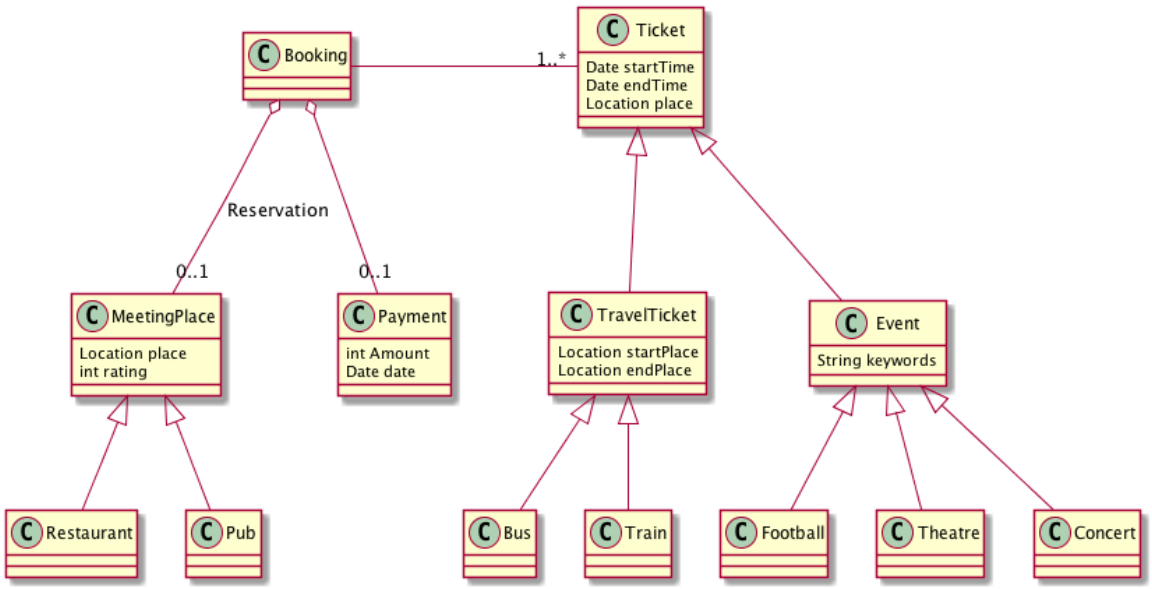
(5) Once the customer have ordered tickets, these are *reserved*. After payment, the tickets are printed and delivered to the customer.

(6) Tickets-R-Us use *Funhouse Ticket Service* to book the tickets and to keep track of available seats. They use *Sardines Travels* to book bus and train tickets. Furthermore, they use *GobGuide* to find pubs and restaurants near the venue.

(7) Payments are done with the help of *Greed Inc.*. Tickets-R-Us gets a message from them whenever a customer has completed a payment. This message also contains information about which reservation the payment concerns.

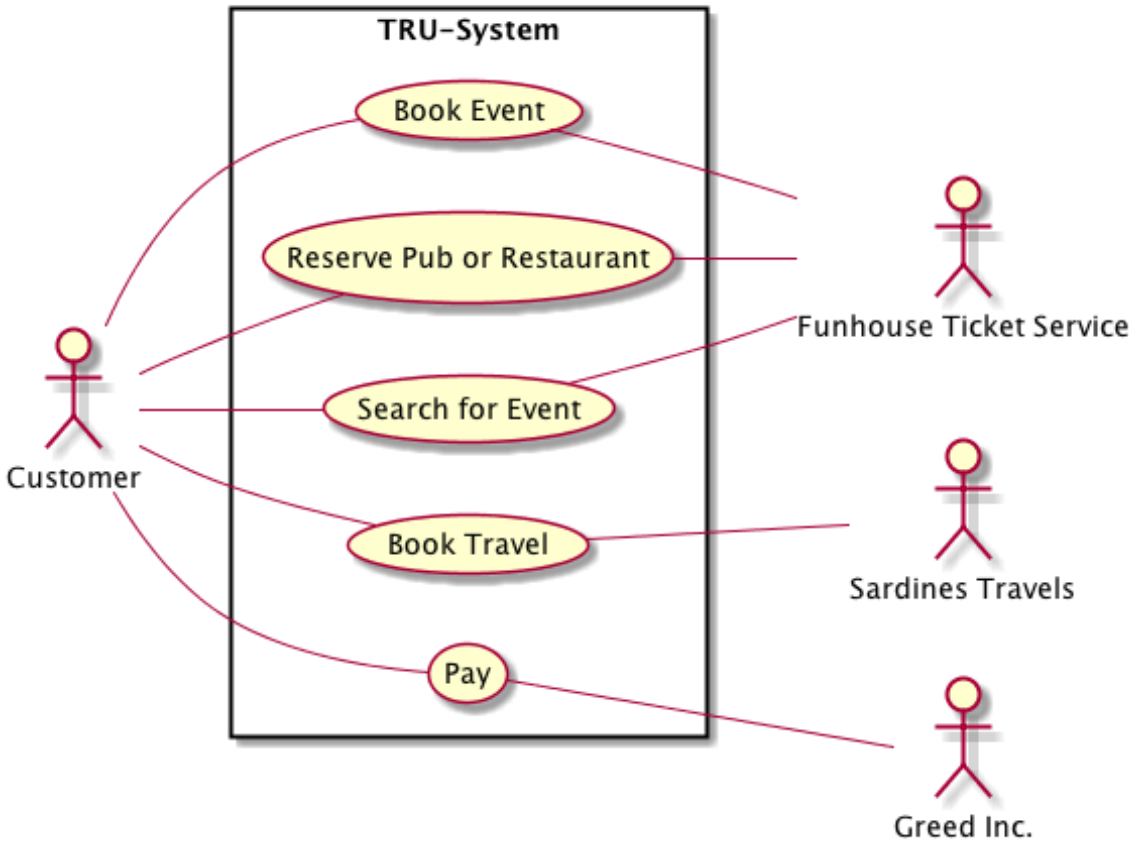
Fråga 6. Konceptuell Modell 6p

Konstruera en *Konceptuell Modell* för det här systemet. Använd associationer, aggregat, arv, och associationsattribut om och när det är tillämpbart. Sätt ut multiplicitet där det är relevant.



Fråga 7. Use Case Diagram 4p

Rita ett *Use Case Diagram* för *TMS*. Se till att du identifierar alla *Aktörer* såväl som alla *Use Cases*. Om du finner det tillämpbart kan du strukturera ditt use case diagram så att några use cases *inkluderar* eller *utökar* andra use cases.



Fråga 8. Expanderat Use Case 4p

Skriv ett *Expanded Use Case* för `Book Known Event`. Det här Use Caset beskriver en kund som vet vilket event hen vill boka (paragraf 3 i beskrivningen ovan).

Use Case: `Book Known Event`

Primary Actor: Customer

Actors: Customer, Funhouse Ticket Service

Preconditions: Customer knows the name of an event, with a start date and time and a location.

Postcondition: The event is booked. Optionally a seat at a nearby restaurant or pub is reserved. Optionally, travel to and from the event is booked.

Brief Description: The customer contacts TRU and want to order a specific event with a specific start date, time, and location. The system confirms that the event is still available and then creates a reservation at the event.

Continued on the next page

Basic Flow:

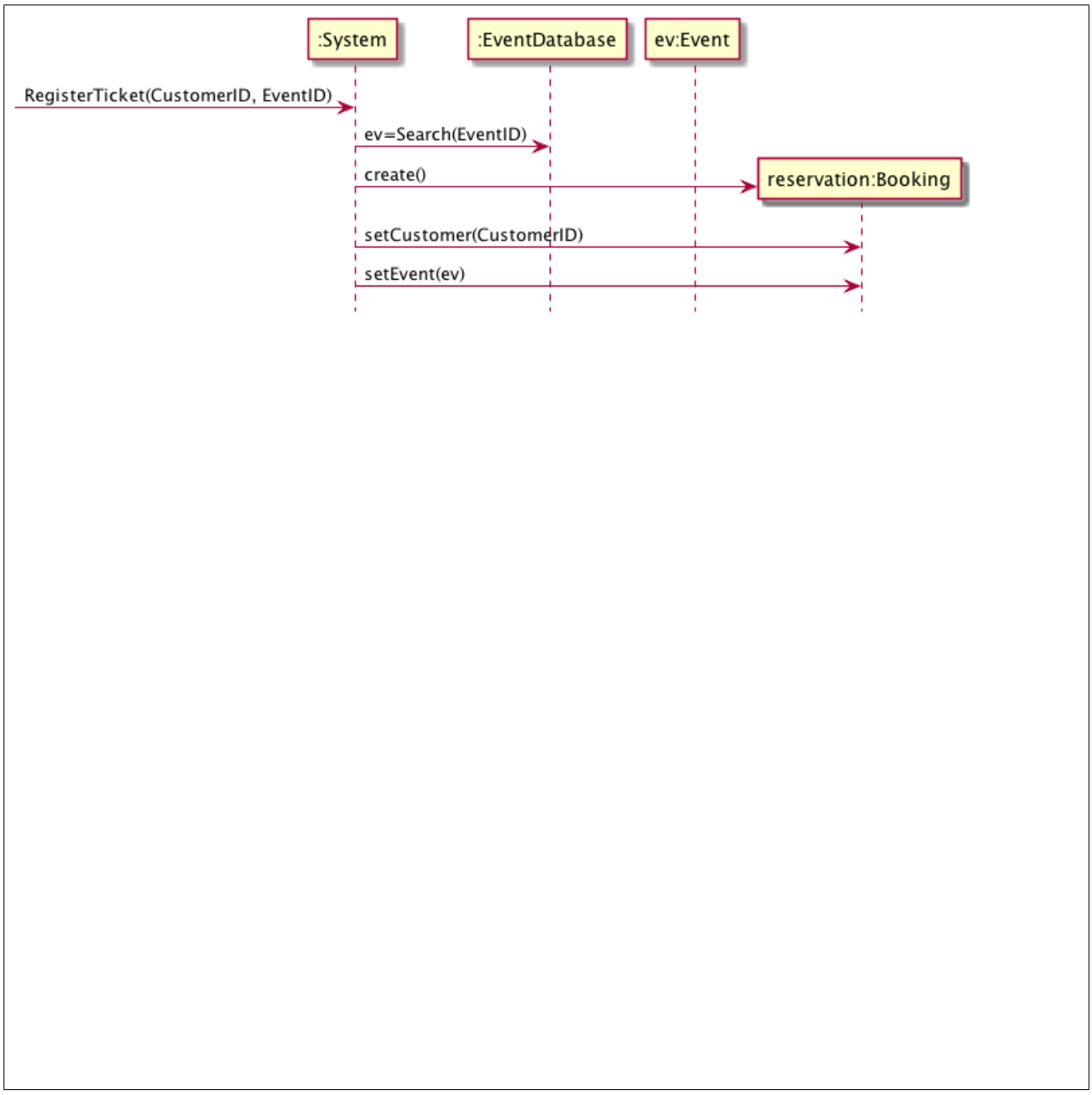
Actor	System
1. The customer enters or selects the name of an event	2. The system shows available dates, times, and locations.
3. The customer selects a date, a time, and a location.	4. The system checks that the event is still available, and creates a reservation.
	5. The system suggests nearby pubs or restaurants. Execute use case Reserve Pub or Restaurant
	6. The system prompts for travel to and from the event. Execute use case Book Travel

Alternative Flows:

- **4.** The event is no longer available at the specified time and place. The system suggests alternatives.

Fråga 9. Interaktionsdiagram 4p

Rita ett *Interaktionsdiagram* (Sekvens- eller kommunikationsdiagram) för systemoperationen `RegisterTicket(CustomerID, EventID)`.



Fråga 10. Klassdiagram 4p

Rita ett klassdiagram som innehåller de klasser, metoder, och attribut som du använde i interaktionsdiagrammet i den förra frågan. Kom ihåg associationer mellan klasserna och lägg till (om och där det är tillämpbart) basklasser och arvshierarkier.