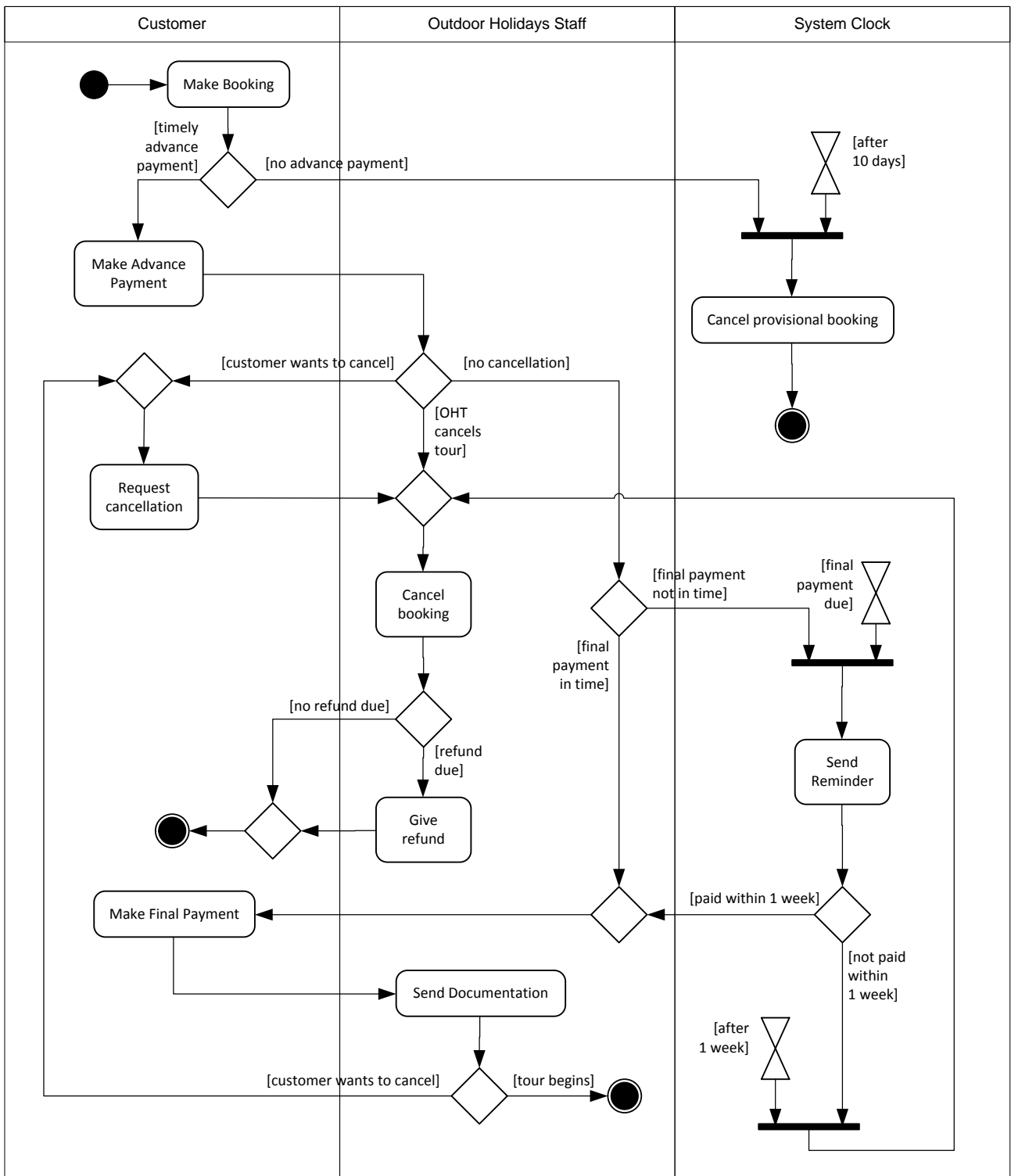# Solutions to the Design Test, 15 December 2016, 8:45–11:45

1. Activity Diagram

Remarks:
- Please note the roles of the different actors in a cancellation.

2a. Glossary

A possible glossary is the following

Term	Description
Participant	Customer of Outdoor Holiday Tours, who takes part in a tour
Footprints	Indicate the difficulty of a tour, ranging from very easy (1) to very challenging (7)
Contact person	When a booking is made for a small group, there is one contact person (whose contact details should be known)

The motivation for choosing terms is that they are important or have a special meaning in the given context. Possible terms, in decreasing order of importance:

- *Participant*: OHT specifically wants this word;
- *Footprints*: an idiosyncratic indication of difficulty, specific for OHT;
- *Contact person*: here specifically for a participant who makes a booking for a small group;
- *Provisional booking*: this is implicity defined in the text (as booking without advance payment);
- *Booking, Cancellation*: not really different from ordinary use, and not surprising that substantial parts of the text are concerned with the details of how OHT handles this;
- *Outdoor Holiday Tours;*
- *Hiking tour, Canoe tour*: no further definition given in the text (but you could list wich aspects are relevant for the system);
- *Advance payment, Final payment*: is used with the standard dictionary meaning;
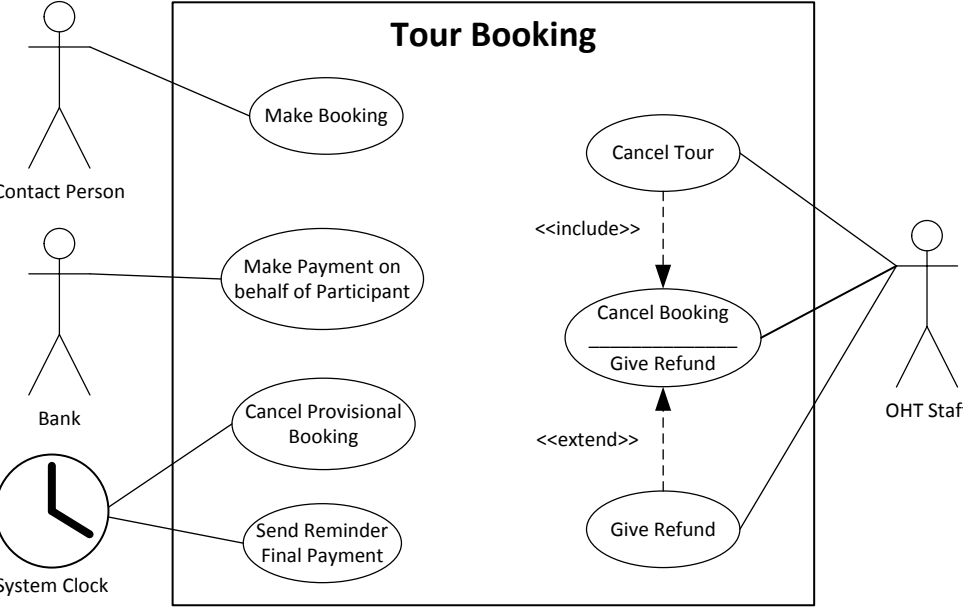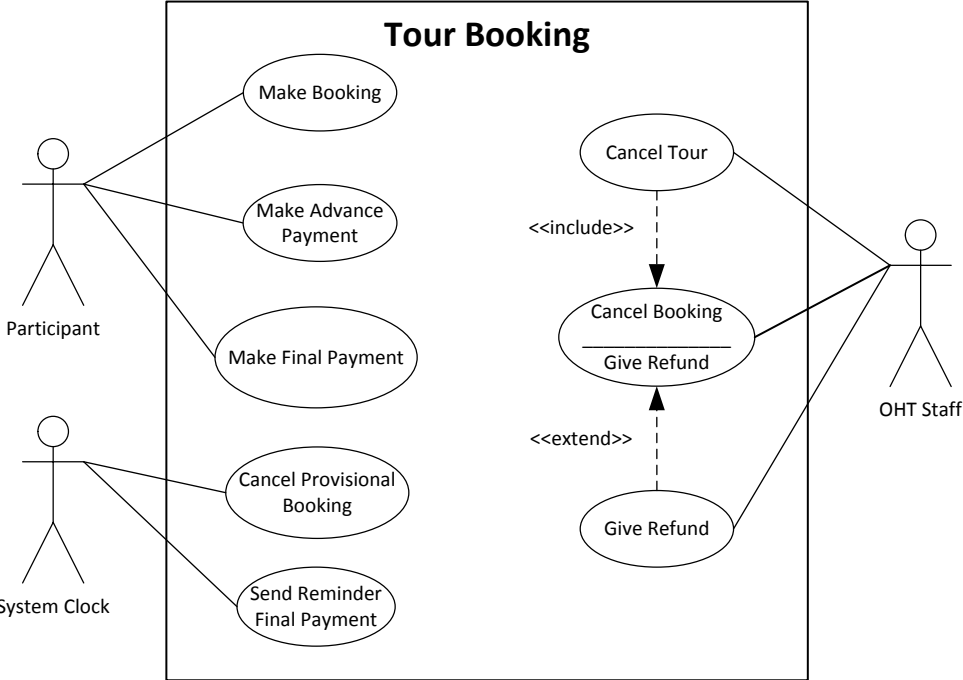- *Diet*: idem.

2b. Use Case Diagram

See the next page for two versions of a use case diagram, each of them fully correct.
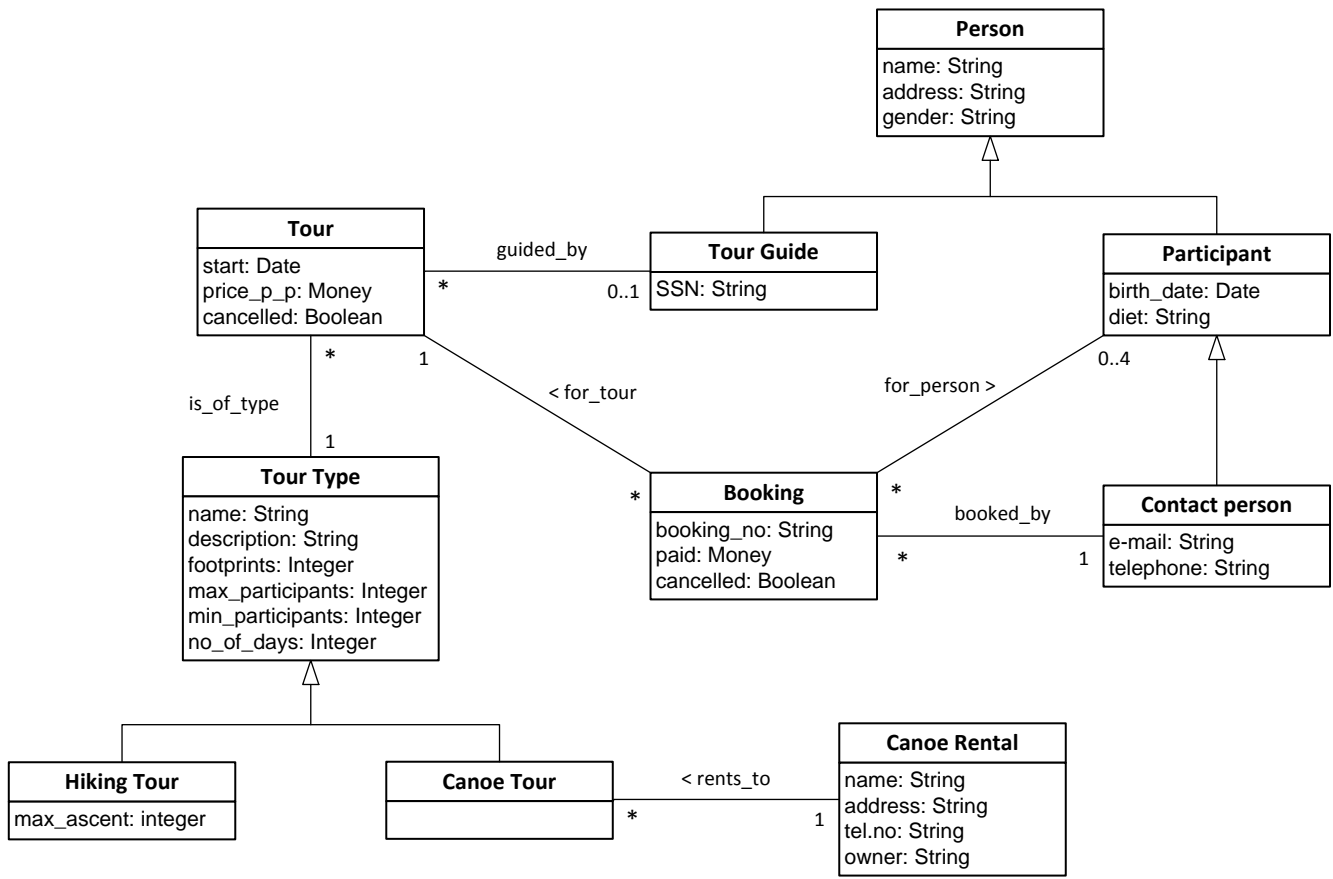
Remarks:

- You could have two different use cases for "Make Advance Payment" and "Make Final Payment", but you can also combine them into a single use case.
- For "Cancel Provisional Booking" and "Cancel Booking" it is convenient to model them as different use cases, each with their own actor.
- For the actor of "Make booking", you can either have "Participant" (where it is assumed that in case of a small group it's the contact person among the participants who makes the booking) or "Contact Person" (where it is assumed that a participant who makes a booking just for himself also can be called contact person).
- Who the actor is for a payment depends on one's perspective. If you want to be very precise, it is probably the bank who makes the payment on behalf of a participant. In a slightly more abstract view (the diagram you'd want to show to OHT when you discuss the system design with them), you could say that the participant makes the payment.
- For "System Clock" you can use a clock symbol (but this notation was not introduced in the slides and exercises); the actor symbol is fine as well.

Two possible Use Case Diagrams

Tour Booking

- Make Booking
- Make Advance Payment
- Make Final Payment
- Cancel Provisional Booking
- Send Reminder Final Payment

Cancel Tour

<<include>>

Cancel Booking
_____
Give Refund

<<extend>>

Give Refund

Participant

System Clock

OHT Staff

Tour Booking

- Make Booking
- Make Payment on behalf of Participant
- Cancel Provisional Booking
- Send Reminder Final Payment

Cancel Tour

<<include>>

Cancel Booking
_____
Give Refund

<<extend>>

Give Refund

Contact Person

Bank

System Clock

OHT Staff

3. Class Diagram

Person
name: String
address: String
gender: String

Tour
start: Date
price_p_p: Money
cancelled: Boolean

guided_by

Tour Guide
SSN: String

Participant
birth_date: Date
diet: String

*   0..1

is_of_type

< for_tour   for_person >   0..4

Tour Type
name: String
description: String
footprints: Integer
max_participants: Integer
min_participants: Integer
no_of_days: Integer

Booking
booking_no: String
paid: Money
cancelled: Boolean

booked_by

Contact person
e-mail: String
telephone: String

*   1

Hiking Tour
max_ascent: integer

Canoe Tour

< rents_to

*   1

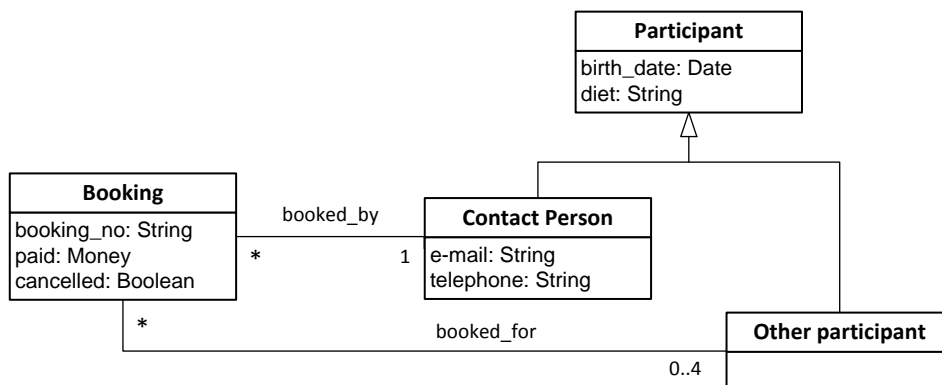**Canoe Rental**
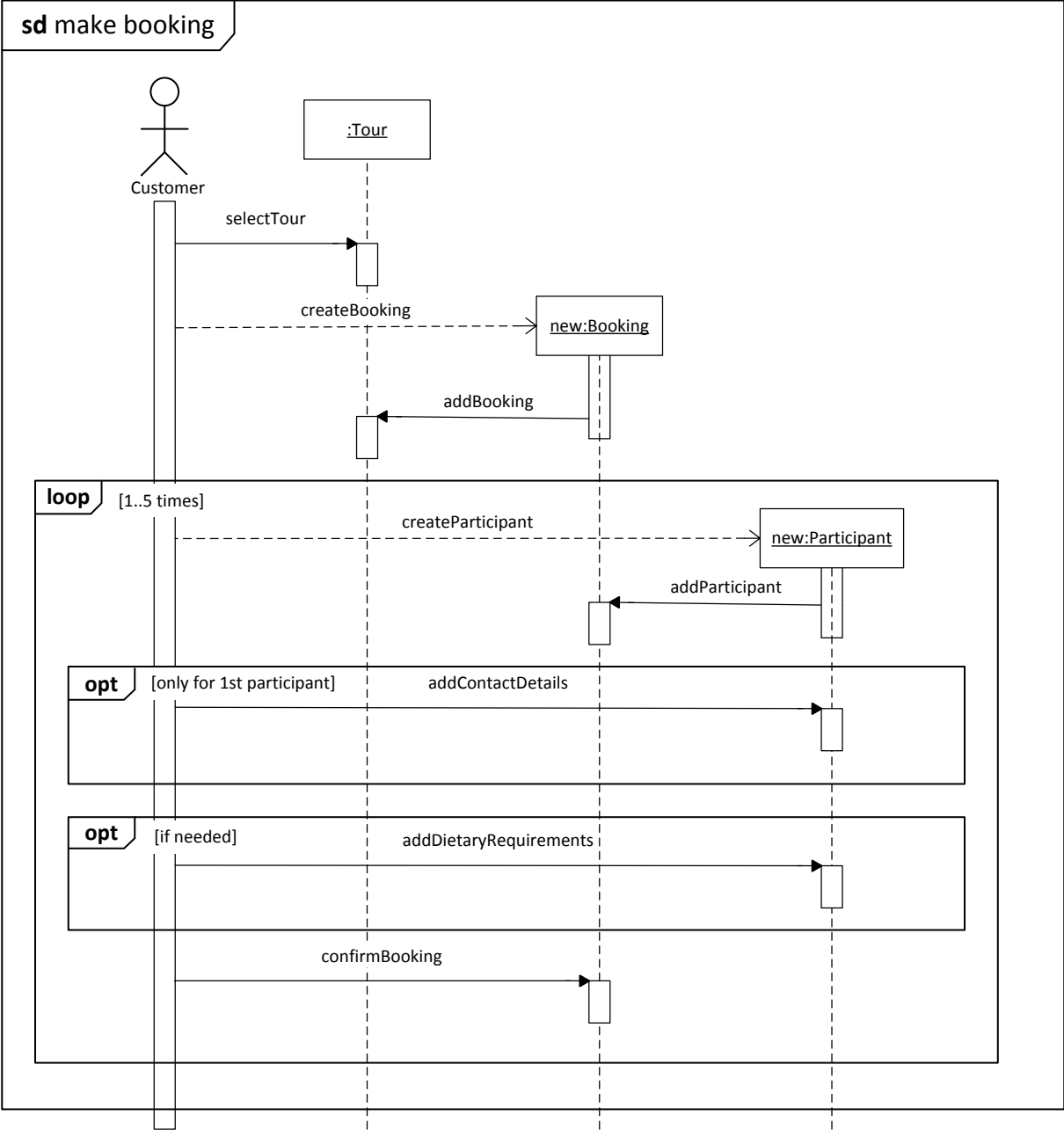name: String
address: String
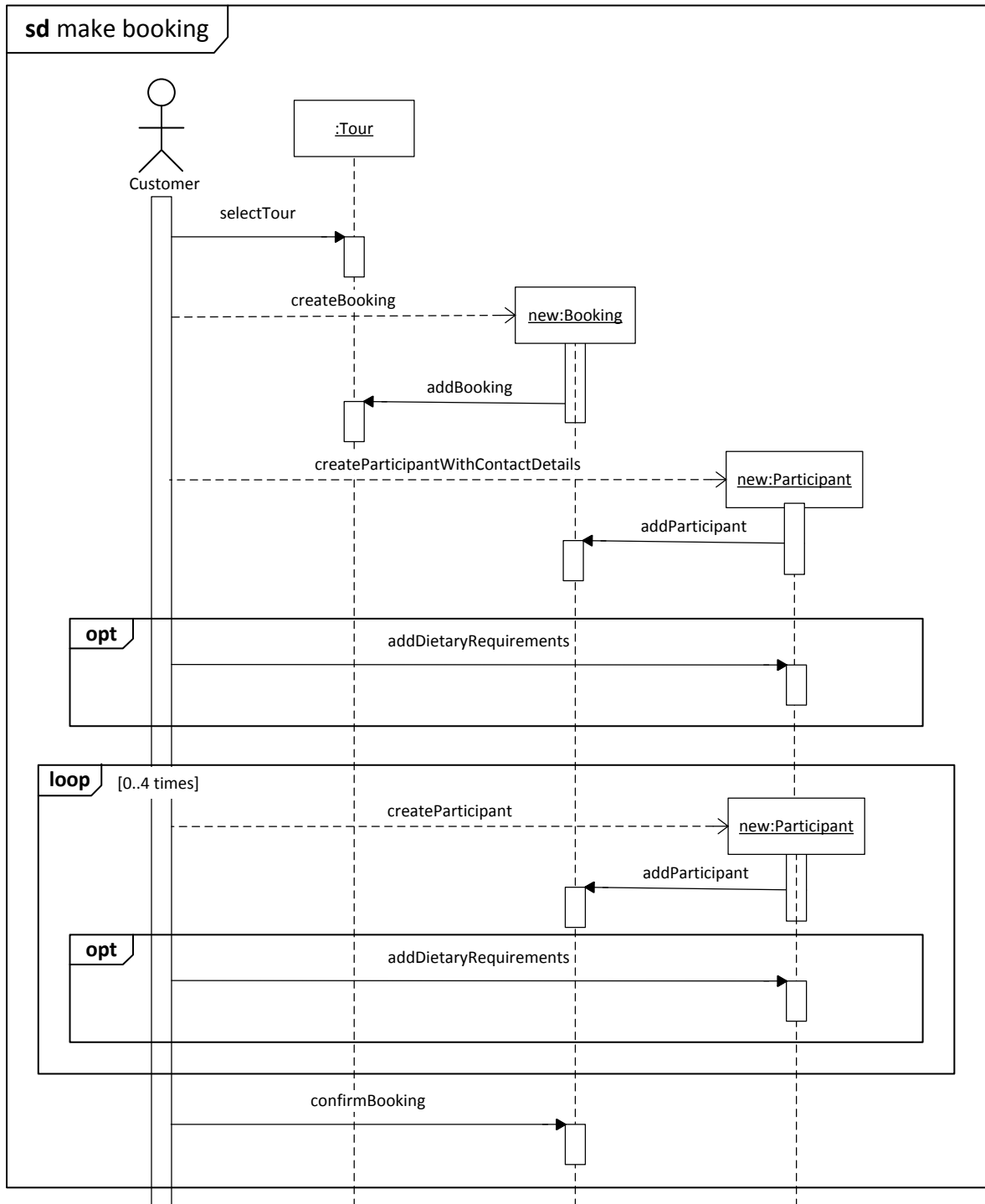tel.no: String
owner: String

Remarks:

- "Tour Type" is not mentioned explicitly, but by now you should be aware of the possible existence of type classes. At various places the text implies that there are different tours of the same type (e.g. "The same tour is more expensive in the peak season")

- It was intended that different tours of a same tour type could have different guides. However, this was not clear from the text. If you have associated Tour Guide with Tour Type, that is considered to be correct as well.

- Participants come in two varieties: contact persons and other participants for one booking. This can be modelled by two subclasses, as shown below. However, the subclass "Other Particpant" is not necessary, as shown in the diagram above.

Participant
birth_date: Date
diet: String

Booking
booking_no: String
paid: Money
cancelled: Boolean

booked_by

Contact Person
e-mail: String
telephone: String

*   1

booked_for

Other participant

0..4

4. Sequence Diagram

There are two equally valid solutions, shown in the following two diagrams. The difference is whether the first participant is taken separately or included in the loop.

sd make booking

Customer

:Tour

selectTour

createBooking → new:Booking

addBooking

loop [1..5 times]

createParticipant → new:Participant

addParticipant

opt [only for 1st participant]   addContactDetails

opt [if needed]   addDietaryRequirements

confirmBooking

sd make booking

Customer

:Tour

selectTour

createBooking

new:Booking

addBooking

createParticipantWithContactDetails

new:Participant

addParticipant

opt

addDietaryRequirements

loop [0..4 times]

createParticipant

new:Participant

addParticipant

opt

addDietaryRequirements
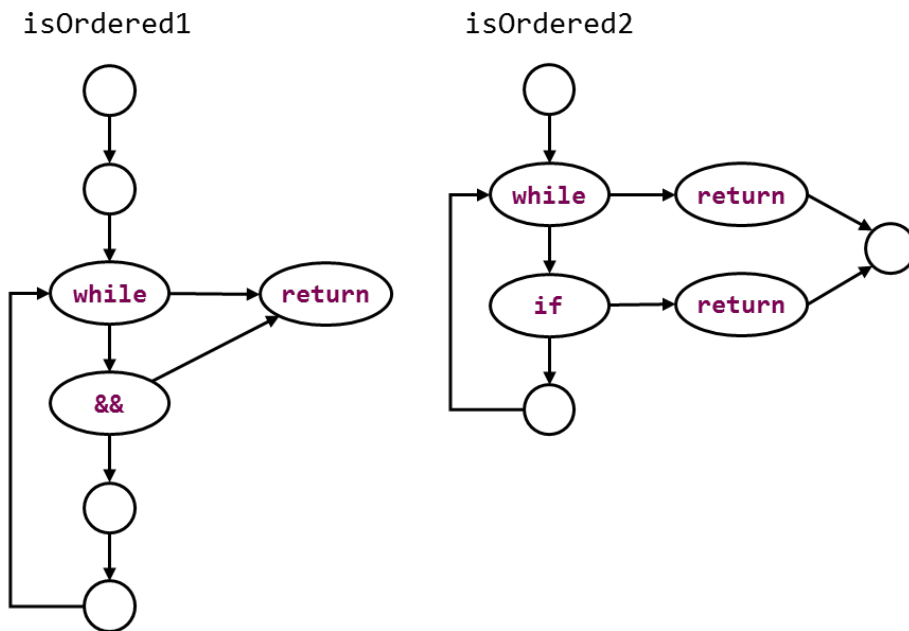
confirmBooking

5a (Software Metrics: Coupling)

1. Counting the constructor, there are 4 similar method pairs (the constructor with each of the getters) and 6 dissimilar ones (the pairs of getters), making for LCOM = 2
2. Each of the classes now has 2 similar method pairs and 1 similar one, making for LCOM = 0

5b (Software Metrics: Cyclometric complexity)

1. The flow graphs:

2. The formula #Edges - #Nodes + 2 yields the same result for both cases

   isOrdered1: $8 - 7 + 2 = 3$
   isOrdered2: $8 - 7 + 2 = 3$

3. Conclusion: the methods are equally complex (in terms of cyclomatic complexity)