Concordia University

Department of Computer Science and Software Engineering

SOEN 331 - S

Formal Methods for Software Engineering

Assignment 2:

Z and Object-Z specifications

Witnick-Hans Joseph

ID: 29348743

Alexandra Zana

ID: 40131077

Alexandre Eid

ID: 40155833

October 28, 2021.

Contents

1 General information

Date posted: Thursday 14 September, 2021.

Date due: Thursday, 28 October, 2021, by 23:59.

Weight: 15% of the overall grade.

2 Introduction

You should form a team of **three** members. Each team should designate a leader who will submit the assignment electronically. In case you cannot find a team, please contact me and I will assign you to one. There are **7** problems in this assignment, with a total weight of **100** points. You must prepare all your solutions in LaTeX and produce a single `pdf` file. Name the file after the Concordia id of the person who will submit, e.g. `123456.pdf`.

3 Ground rules

This is an assessment exercise. You may not seek any assistance while expecting to receive credit. **You must work strictly within your team and seek no assistance for this assignment ((e.g. from the teaching assistants, fellow classmates and other teams or external help)**. Please note that you should **not** discuss the assignment during tutorials. I am available to discuss clarifications in case you need any.

All team members are expected to work relatively equally on each problem. The team leader has the responsibility to ensure that the team does not violate this rule. **In your submission, you must include only the names of those team members who contributed to the assignment.** Accommodating someone who did not contribute will result in a penalty.

If there is any problem in the team (such as lack of contribution, etc.), the team leader must contact the instructor as soon as the problem appears.

4 Problems

4.1 Part 1 (40 pts): Temperature monitoring system with the Z specification

Consider a system called 'TempMonitor' that keeps a number of sensors, where each sensor is deployed in a separate location in order to read the location's temperature. Before the system is deployed, all locations are marked on a map, and each location will be addressed by a sensor. The formal specification of the system introduces the following three types:

SENSOR TYPE, LOCATION TYPE, TEMPERATURE TYPE

We also introduce an enumerated type MESSAGE which will assume values that correspond to success and error messages.
Provide a formal specification in Z, with the following operations:

- **DeploySensorOK**: Places a new sensor to a unique location. You may assume that some (default) temperature is also passed as an argument.
- **ReadTemperatureOK**: Obtain the temperature reading from a sensor, given the sensor's location.

Provide appropriate success and error schemata to be combined with the definitions above to produce robust specifications for the following interface:

- **DeploySensor**,
- **ReadTemperature**

4.2 Part 2 (60 pts): A booking system with the Object Z specification

We introduce the basic types [Person, SeatType]. We also introduce an enumerated type Message which will assume values (feel free to define your own) that correspond to success and error messages. Consider a system to book seats for a theater play. A customer can book a single seat, and a seat can only accommodate a single customer. The booking system keeps a log of the customers that have booked a seat. The system publishes a plan of the theater and it allows customers to access it online and make a booking or cancel a booking.

4.2.1 Class Booking

Define a formal specification in Object-Z for class Bookingt to support the following operations:

- **BookOK**: Reserves a seat for a given customer.
- **CancelOK**: Frees a seat for a given customer.

You will also need to provide appropriate success and error schemata to be combined with the definitions above to produce robust specifications for the following interface:

- **Book**, and
- **Cancel**.

4.2.2 Class Booking2

Subclassify Booking to introduce class Booking2 that behaves exactly like Booking, while introducing the following operations:

- **GetNumberOfCustomers** returns the total number of customers who have made a booking.
- **ModifyBookingOK** assigns an existing customer to a different seat. Provide any additional schema(ta)
  in order to extend the interface to include a robust operation ModifyBooking.

The extended interface will now include operations

- **GetNumberOfCustomers**, and
- **ModifyBooking**.

5 Object Z

Booking

$\upharpoonright (Book, Cancel)$

$$
\begin{array}{|l}
reserved : \mathcal{P}\, Person \\
book : Person \pfun SeatType \\
capacity : \mathbb{N} \\
count : \mathbb{N} \\
\hline
reserved = dom\ book \\
capacity > 0 \\
count \geq 0 \\
\hline
\end{array}
$$

Init

$$
\begin{array}{|l}
book = \varnothing \\
capacity = \mathbb{N} \\
count = 0 \\
\hline
\end{array}
$$

BookOk

$$
\begin{array}{|l}
\Delta(book, count) \\
customer? : Person \\
seat? : SeatType \\
\hline
customer? \notin reserved \\
seat? \notin ran\ book \\
count < capacity \\
book' = book \cup \{customer? \mapsto seat?\} \\
count' = count + 1 \\
\hline
\end{array}
$$

CancelOk

$$
\begin{array}{|l}
\Delta(book, count) \\
customer? : Person \\
\hline
customer? \in reserved \\
count > 0 \\
book' = \{customer?\} \rhd book \\
count' = count - 1 \\
\hline
\end{array}
$$

CustomerExists

$$
\begin{array}{|l}
customer? : Person \\
response! : Message \\
\hline
customer? \in reserved \\
response! = \ 'Customer\ Exists' \\
\hline
\end{array}
$$

...

Map

...

CustomerNotFound

$customer? : Person$

$response! : Message$

$customer? \notin reserved$

$response! = \text{'Customer Not Found'}$

SeatUnavailable

$seat? : SeatType$

$response! : Message$

$seat? \in ran\ book$

$response! = \text{'Seat Is Already Taken'}$

TheaterEmpty

$response! : Message$

$count = 0$

$response! = \text{'Room Is Empty'}$

FullyBooked

$response! : Message$

$count = capacity$

$response! = \text{'Room is Full'}$

Success

$response! : Message$

$response! = \text{'Ok!'}$

$Book \mathrel{\hat{=}} (BookOk \land Success) \oplus CustomerExists \oplus (SeatUnavailable \lor FullyBooked)$

$Cancel \mathrel{\hat{=}} (CancelOk \land Success) \oplus (CustomerNotFound \lor TheaterEmpty)$

6 What to submit

Please submit your `pdf` file at the Electronic Assignment Submission portal

(`https://fis.encs.concordia.ca/eas`)

under **Theory Assignment 2**.