Prof. Dr. Peter Thiemann
Sergio Feo-Arenis
Sergiy Bogomolov

Summer Term 2014

# Software Engineering

http://proglang.informatik.uni-freiburg.de/teaching/swt/2014/

## Exercise Sheet 2

### Exercise 1 (20 points)

In the lecture, you have seen an approach to formally define requirements by using before-after predicates as shown below:

enterBuilding(p) $\triangleq$
PRE
$hasAuthorization(p) \wedge p \in carriesPassport$
THEN
$peopleInBuilding := peopleInBuilding \cup \{p\} \parallel$
$passportsAtDesk := passportsAtDesk \cup \{p\} \parallel$
$carriesPassport := carriesPassport - \{p\}$
END

In this exercise, you need to formalize using this notion the specification of a lift which works as follows:

1. The lift has a set of buttons inside of the cabin corresponding to each floor. Buttons in the lift are illuminated if pressed to provide the information about the floors still to be visited.
2. Each floor has a button to request the lift. Buttons on the floors are illuminated if pressed.
3. Lift door must be closed when the lift is moving.
4. The building has a number of "secure" floors *SECURE* accessible by only authorized personnel. In order to go to a secure floor, a user has to insert an appropriate key into the lock and press a button with the required floor number afterwards.

Below you find formal specifications of the following operations:

1. *request_lift*($f$) - the user requests the lift at floor $f$.
2. *request_floor*($f$) - the user requests to stop the lift at floor $f$.
3. *depart*($dir$) - the lift can start moving in the direction $dir$ where $dir \in \{ up, down\}$.
4. *arrive*($f$) - the lift must stop at the floor $f$.

$floor \in INTEGER$
$moving \in BOOL$
$door \in \{open, closed\}$

request_lift($f$) $\triangleq$
PRE
$f \in FLOORS$
THEN
$call\_requests := call\_requests \cup \{f\}$
END

request_floor($f$) $\triangleq$
PRE
$f \in FLOORS$
THEN
$stop\_requests := stop\_requests \cup \{f\}$
END

depart ($dir$) $\triangleq$
PRE
$dir \in \{up, down\} \wedge moving = false \wedge door = open$
THEN
$moving := true \parallel$
$floor := next(dir, floor)$
END

arrive (f) $\triangleq$
PRE
$f \in FLOORS \wedge moving = true \wedge req\_floor(f) \wedge door = closed$
THEN
$call\_requests := call\_requests - \{f\}$
END

Check the formal specification for incompleteness, internal inconsistencies as well as inconsistencies with the informal specification. Can you identify situations where the elevator control gets stuck or where the informal specification is violated? You may also suggest changes to the informal specification, if it does not make sense.

Hint: There are also invariants on the state of the elevator.

The formal specification relies on the following auxiliary definitions (functions and predicates):

1. $key\_inserted() = true$ if an appropriate key is in the lock.
2. $requests(dir) = true$ if they are floor requests in the direction $dir$.
3. $next(dir, floor)$ is equal to the next floor where the lift has to stop in the direction $dir$ when going from $floor$.
4. $req\_floor(floor) = true$ if the lift needs to stop at $floor$.

Provide formal definitions for $request(dir)$, $next(dir, floor)$, and $req\_floor(floor)$ as functions or predicates on the state of the elevator control.

## Solution

$MAX\_FLOOR \in INTEGER$
$FLOORS \in 0 \ldots MAX\_FLOOR$
$floor \in FLOORS$
$moving \in BOOL$
$door \in \{open, closed\}$
$call\_requests \in set\ of\ INTEGER$
$stop\_requests \in set\ of\ INTEGER$

$INVARIANT : moving = true \Leftrightarrow door = closed$

request_lift($f$) $\triangleq$
PRE
$f \in FLOORS$
THEN
$call\_requests := call\_requests \cup \{f\}$
END

request_floor($f$) $\triangleq$
PRE
$f \in FLOORS \wedge (f \in SECURE \rightarrow key\_inserted())$
THEN
$stop\_requests := stop\_requests \cup \{f\}$
END

depart ($dir$) $\triangleq$
PRE
$dir \in \{up, down\} \wedge moving = false \wedge door = open \wedge requests(dir)$
THEN
$door := closed \parallel$
$moving := true \parallel$
$floor := next(dir, floor)$
END

arrive (f) $\triangleq$
PRE
$f \in FLOORS \wedge moving = true \wedge req\_floor(f) \wedge door = closed$
THEN
$moving := false \parallel$
$door := open \parallel$
$call\_requests := call\_requests - \{f\} \parallel$
$stop\_requests := stop\_requests - \{f\}$
END

The informal specification can be refined as follows:

1. All requests for the lift from floors must be satisfied eventually.
2. All requests for floors within the lift must be satisfied eventually.

The formal definitions for $request(dir)$, $next(dir, floor)$, and $req\_floor(floor)$ as functions or predicates on the state of the elevator control can be written as follows:

1. $req\_floor(floor) = floor \in stop\_requests \vee floor \in call\_requests$

2. $request(dir) = (dir = down \rightarrow \exists\, 0 \leq f < floor : req\_floor(f)) \wedge$
$(dir = up \rightarrow \exists\, floor \leq f < MAX\_FLOOR : req\_floor(f))$

3. $next(dir, floor) = \begin{cases} \max\limits_{0 \leq f < floor} \{f \mid req\_floor(f)\}, & \text{if } dir = down \\ \min\limits_{floor \leq f < MAX\_FLOOR} \{f \mid req\_floor(f)\}, & \text{if } dir = up \end{cases}$

---

**Submission**

- Submit this sheet *before* the lecture of Thursdays.
- Late submissions will not be accepted.
- Deadline: Thursday 11:59 a.m..