

Formal Methods

M5X01036

Coursework

SA1 Site Safety System

completed by

Eben Stewart

P115266

for Lecturer

Gordon Dickers

10 May 2018

1 Introduction

A system to manage staff members on the site of Steve's Friendly Nuclear Company. The Company has an access controlled site containing three secure, access controlled work areas. Due to the nature of the business, the company strictly controls access to the work areas. In this context On_Site indicates that staff are between buildings of the power plant. One can leave the site only through the Off_Site state. To enter a work area a staff member must log in. To leave it s/he must log out.

There are four security levels. Staff have access to a work area based on their security level within the company. These relationships are defined in the table below:

	Reactor_Core	Control_Room	Tech	On_Site	Off_Site
Tech		x	x	x	x
Science	x	x		x	x
Management		x	x	x	x
Office_Staff			x	x	x

2 Types and Globals

Here we will define some freetypes and globals.

[STAFF] The set of all uniquely identifiable staff of the company

A person working at the company:

staff : $\mathbb{P}$ STAFF

Freetypes:

WORK_AREA::=

Reactor_Core | *Control_Room* | *Admin_Building* | *On_Site* | *Off_Site*

SECURITY_LEVEL::= *Tech* | *Science* | *Management* | *Office_Staff*

The following messages can be produced by the system:

MESSAGE::= *staff_entered* | *staff_exited* | *staff_added* | *staff_removed* | *staff_not_present* | *staff_in_area* | *staff_not_in_area* | *staff_on_site* | *staff_not_on_site* | *access_violation* | *already_Staff* | *not_staff* | *security_level_not_null* | *logged_in* | *area_changed* | *area_unchanged*

Relationships:

The following statements in set notation define the relationships between work areas:

$$\begin{array}{|l} \{Reactor_Core, Control_Room, Admin_Building\} \subset On_Site \\ Off_Site \notin On_Site \end{array}$$

The following relations describe the relationships between SECURITY_LEVEL and WORK_AREA. This relation defines which security level confers access to which work areas.

$$\begin{array}{|l} canAccess : SECURITY_LEVEL \leftrightarrow WORK_AREA \end{array}$$

3 The State of the System

$\begin{array}{l} \text{NukePlant} \\ \hline staff : \mathbb{P} STAFF \\ allowedArea : \mathbb{P} WORK_AREA \\ disallowedArea : \mathbb{P} WORK_AREA \\ currentArea : \mathbb{P} WORK_AREA \\ newArea : \mathbb{P} WORK_AREA \\ _hasSecurityLevel_ : STAFF \leftrightarrow SECURITY_LEVEL \\ _loggedIn_ : STAFF \rightarrow WORK_AREA \end{array}$
$\begin{array}{l} staff = \text{dom } _hasSecurityLevel \\ allowedArea = \text{ran } canAccess \\ disallowedArea : WORK_AREA \setminus allowedArea \\ currentArea \in allowedArea \\ newArea \in allowedArea \\ \exists_1 currentArea \mid s \text{ loggedIn } currentArea \bullet s \in staff \\ \exists_1 newArea \mid s \text{ loggedIn } newArea \bullet s \in staff \end{array}$

4 Initialisation

$\begin{array}{l} \text{InitNukePlant} \\ \hline \text{NukePlant}' \\ \hline staff' = \emptyset \\ loggedIn' = \emptyset \\ hasSecurityLevel' = \emptyset \\ currentArea' = \emptyset \\ newArea' = \emptyset \\ canAccess' = \{ Tech \mapsto Control_Room, Tech \mapsto Admin_Building, \\ Tech \mapsto On_Site, Tech \mapsto Off_Site, Science \mapsto Reactor_Core, \\ Science \mapsto Control_Room, Science \mapsto Admin_Building, \\ Science \mapsto On_Site, Science \mapsto Off_Site, Management \mapsto Control_Room, \\ Management \mapsto Admin_Building, Management \mapsto On_Site, \\ Management \mapsto Off_Site \} \end{array}$
--

5 Change State Operations

Operations defined here are:

- i) Add staff
- ii) Remove staff
- iii) ChangeArea
- iv) Leave Area
- v) Is staff in Area
- vi) Is staff On Site

These schema are composed of *partial* operations which are then developed into *total* operations by considering pre and post error conditions.

5.1 Add Staff

Adds a new staff member. This is composed of two schema $AddStaff_0$ and $AddStaffError$. The assignment brief seems somewhat ambiguous. Add Staff could be taken to mean either add an existing staff member to an area or it could mean add (hire) a new staff member to the company. Here it shall be interpreted to mean the latter. Likewise, the schema Remove Staff (below) will be taken to mean to remove a member of staff from the company altogether, not just a work area.

$AddStaff_0$ $\Delta NukePlant$ $s? : STAFFREPLY! : MESSAGE$
$s? \notin staff$ $staff' = staff \cup \{s?\}$ $loggedIn' = loggedIn$ $hasSecurityLevel' = hasSecurityLevel$ $currentArea' = allowedArea \mid s? \in \text{dom } loggedIn$ $reply! = staff_added$

$AddStaffError$ completes the AddStaff operation by specifying how to deal with any errors.

$AddStaffSError$ $\Xi NukePlant$ $s? : STAFF$ $reply! : MESSAGE$
$S? \in STAFF$ $reply! = already_staff$

Disjunction of the two schemas completes the AddStaff schema:
 $AddStaff \Leftrightarrow AddStaff_0 \vee AddStaffError$

5.2 Remove Staff

Removes a staff member. $s?$ must be a staff, must not have a Security Level or be logged in to a work area. No other state changes to occur. As above, the total schema will be composed of $RemoveStaff_0$ disjoined with $RemoveStaffError$.

$RemoveStaff_0$	_____
$\Delta NukePlant$ $s? : STAFF$	
$s? \in staff$ $staff' = staff \setminus \{s?\}$ $loggedIn' = s? \triangleleft loggedIn$ $hasSecurityLevel' = s? \triangleleft hasSecurityLevel$ $currentArea' = \emptyset$	

$RemoveStaffError$	_____
$\exists NukePlant$ $s? : STAFF$ $reply! : MESSAGE$	
$s? \notin staff \wedge reply! = not_staff$ $\vee$ $s? \in staff \wedge s? \in \text{dom } hasSecurityLevel \wedge response! = security_level_not_null$ $\vee$ $s? \in staff \wedge s? \in \text{dom } loggedIn \wedge response! = logged_in$	

$$RemoveStaff \Leftrightarrow RemoveStaff_0 \vee RemoveStaffError$$

5.3 Change Area

This specifies that $s?$ will change work area to $a?$ but $s?$ must be logged into a work area already. The new work area must not be the same as the old work area.

$ChangeArea_0$ $\Delta NukePlant$ $s? : STAFF$ $a? : Work_area$ $reply! : MESSAGE$
$s? \in staff$ $a? \neq currentArea$ $s? \text{ loggedIn } currentArea \Rightarrow s? \in currentArea$ $currentArea' = s? \Leftarrow currentArea$ $loggedIn' = loggedIn \bullet \text{dom } loggedIn \setminus s?$ $newArea = a?$ $newArea' = newArea \cup s? \Rightarrow s? \text{ loggedIn } newArea \Rightarrow$ $loggedIn'' = loggedIn' \cup \{s? \mapsto newArea\}$ $currentArea'' = newArea'$ $reply! = area_changed$

$changeAreaError$ completes the $changeArea$ operation by specifying how to deal with any errors.

$ChangeAreaError$ $\Xi NukePlant$ $s? : STAFF$ $a? : WORK_AREA$ $reply! : MESSAGE$
$s? \notin staff \wedge reply! = not_staff$ $\vee$ $s? \in staff \wedge s? \text{ loggedIn } a? \wedge reply! = area_unchanged$

Disjunction of the two schemas completes the $changeArea$ schema:
 $changeArea \Leftrightarrow changeArea_0 \vee changeAreaError$

5.4 Leave Area

this specifies that $s?$ will leave area $a?$. Preconditions are that $s?$ must already be a staff and logged in to the area $s?$ will leave.

Since `off_site` is actually defined as a work area in the assignment brief and implemented as such here, it needs to be logged in to just the same as the other work areas. Thus the idea of leaving an area without logging into another makes no sense and cannot be implemented.

Instead, an alternative schema `Leave Site` will be implemented. Even this schema, in this author's mind, superfluous as the same effect is achieved by calling the `changeArea` schema and setting $a?$ to `off_site`. So:

5.5 Leave Site

$leaveSite_0$
$a? \neq currentArea$ $s? \text{ loggedIn } currentArea \Rightarrow s? \in currentArea$ $currentArea' = s? \triangleleft currentArea$ $loggedIn' = loggedIn \bullet \text{dom } loggedIn \setminus s?$ $on_site = on_site \cup \{s?\} \Rightarrow s? \text{ loggedIn } on_site$ $on_site' = s? \triangleleft on_site$ $loggedIn' = loggedIn \bullet \text{dom } loggedIn \setminus s?$ $off_site = off_site \cup \{s?\} \Rightarrow s? \text{ loggedIn } off_site$
$leaveSiteError$
$\exists NukePlant$ $s? : STAFF$ $a? : WORK_AREA$ $reply! : MESSAGE$
$s? \notin staff \wedge reply! = not_staff$ $\vee$ $s? \in staff \wedge s? \text{ loggedIn } off_site \wedge reply! = area_unchanged$

Disjunction of the two schemas completes the `changeArea` schema:

$$leaveSite \Leftrightarrow leaveSite_0 \vee leaveSiteError$$

5.6 Is Staff In Area

For this schema, the precondition are that the person in question ($s?$) is a valid staff member. The schema will test if $s?$ is logged in to work area $a?$. No error schema is provided.

$isStaffInArea$ $\Delta NukePlant$ $s? : STAFF$ $a? : WORK_AREA$ $reply! : MESSAGE$	
$s? \in staff$ $\exists_1 reply! \mid$ $s? \in a? \wedge reply! = staff_in_area$ $\vee$ $s? \notin a? \wedge reply! = staff_not_in_area$	

5.7 Is Staff On Site

For this schema, the precondition are that the person in question ($s?$) is a valid staff member. The schema tests whether a staff member is on or off site. No error schema is provided.

$isStaffOnSite$ $\Delta NukePlant$ $s? : STAFF$ $reply! : MESSAGE$	
$s? \in staff$ $\exists_1 reply! \mid$ $s? \notin off_site \wedge reply! = staff_on_site$ $\vee$ $s? \in off_site \wedge reply! = staff_not_on_site$	