Development of Control Systems Guided by Models of their Environment

Simon Hudon and Thai Son Hoang

Chair of Information Security, Department of Computer Science
Swiss Federal Institute of Technology Zürich (ETH Zürich)

B Workshop, Limerick, Ireland
21st June 2011

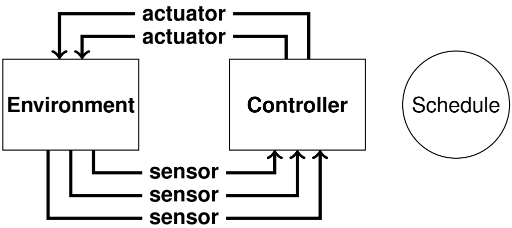
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Event-B Modelling Method

Event-B can be used to model:

- distributed systems,
- concurrent systems,
- sequential programs,
- control systems,
- etc.

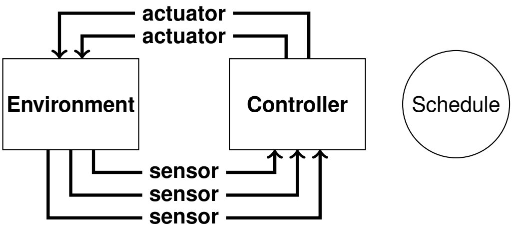
More on Formalising Control Systems



- Controller interacts with its environment via sensors/actuators.
- Event-B models complete system, including environment.
- Greater complexity (compared to models of the controller alone).
- Some existing examples, e.g. in Abrial's Event-B book.
- Developing control systems in Event-B
  remains an art rather than an engineering discipline.

ETH Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

A Modelling Guideline



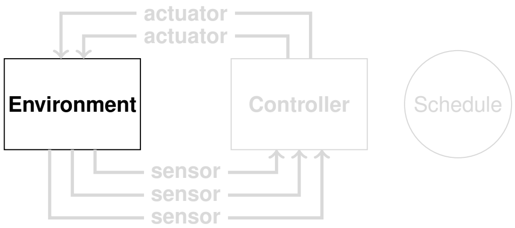
Stage 1 To model the environments as it should behave.

Stage 2 To model the actuators to command environment's changes.

Stage 3 To model the sensors together with the controller.

Stage 4 To model some appropriate scheduler for the controller.

A Modelling Guideline



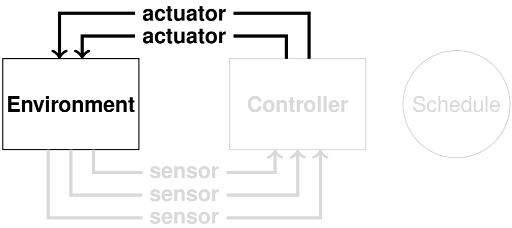
Stage 1 To model the environments as it should behave.

Stage 2 To model the actuators to command environment's changes.

Stage 3 To model the sensors together with the controller.

Stage 4 To model some appropriate scheduler for the controller.

A Modelling Guideline



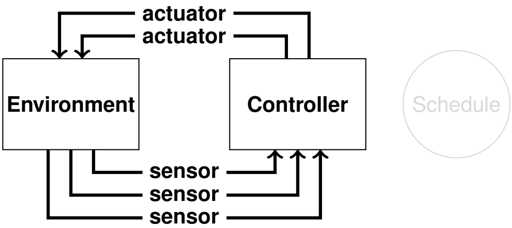
Stage 1 To model the environments as it should behave.

Stage 2 To model the actuators to command environment's changes.

Stage 3 To model the sensors together with the controller.

Stage 4 To model some appropriate scheduler for the controller.

A Modelling Guideline



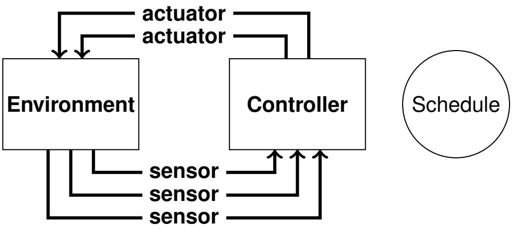
Stage 1 To model the environments as it should behave.

Stage 2 To model the actuators to command environment's changes.

Stage 3 To model the sensors together with the controller.

Stage 4 To model some appropriate scheduler for the controller.

A Modelling Guideline



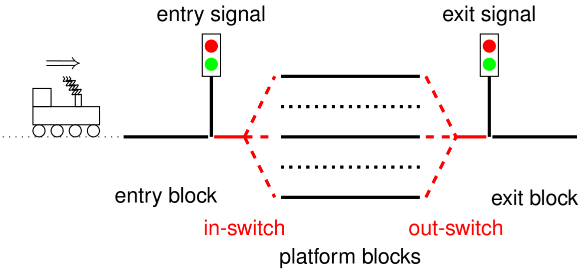
Stage 1 To model the environments as it should behave.

Stage 2 To model the actuators to command environment's changes.

Stage 3 To model the sensors together with the controller.

Stage 4 To model some appropriate scheduler for the controller.

Signal Control at a Stations

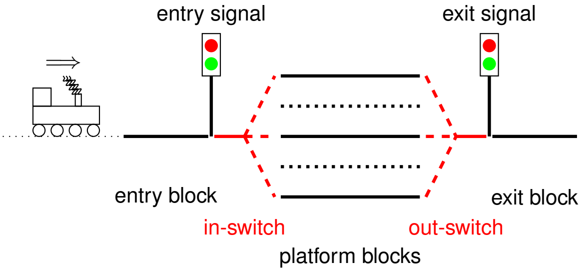


ENV0 There are platforms in between an entry block and an exit block.

ENV1 A train occupies no more than one block.

ENV2 The track is one-way.

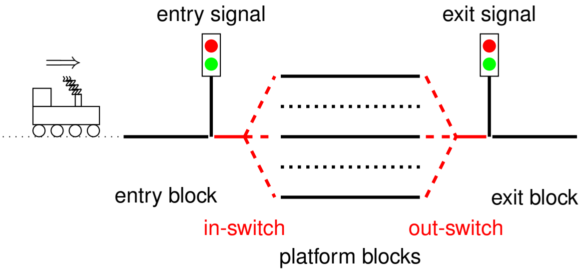
Environment



ENV3 There are switches connecting the entry/exit block to a platform.

ENV4 A train at entry block can only enter/leave some platform block if the in/out-switch is set to that particular block.

Safety Requirement



SAF5 Two trains cannot be on the same block.

ENV6 There are two signals which are either red or green.

ENV7 Trains are assumed to stop at red signals.

Sensors and Actuators

ENV8 There are sensors detecting whether a block is occupied.

ENV9 There are sensors detecting the status of the signals.

ENV10 The sensors reflect the current status of the components.

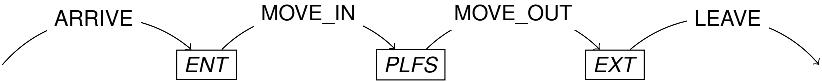
ENV11 For each signal, there is an actuator for the controller to command the signal to turn from red to green.

ENV12 The signals change from green to red when a train passes by.

ENV13 For each switch, there is an actuator for the controller to command the switch to change to a specific platform.

Stage 1. To Model the Environment (1/4)

The "Occupied" Blocks



variables: OCC

invariants:
$OCC \subseteq BLOCK$

ARRIVE
when
$ENT \notin OCC$
then
$OCC := OCC \cup \{ENT\}$
end

MOVE_IN
any p **where**
$p \in PLFS \setminus OCC$
$ENT \in OCC$
then
$OCC := (OCC \cup \{p\}) \setminus \{ENT\}$
end

Stage 1. To Model the Environment (2/4)

The Switches

variables: $\ldots, IN\_SW, OUT\_SW$

invariants:
$IN\_SW, OUT\_SW \in PLFS$

(abstract_) MOVE_IN
any p **where**
$p \in PLFS \setminus OCC$
$\ldots$
then
$OCC := (OCC \cup \{p\}) \setminus \{ENT\}$
end

(concret_) MOVE_IN
when
$IN\_SW \notin OCC$
$\ldots$
with
$p = IN\_SW$
then
$OCC := (OCC \cup \{IN\_SW\}) \setminus \{ENT\}$
end

- ARRIVE, LEAVE are unchanged.
- TURN_IN_SW: new event.

TURN_IN_SW
begin
$IN\_SW :\in PLFS$
end

Stage 1. To Model the Environment (3/4)

The Signals

variables: $\ldots, ENT\_SGN, EXT\_SGN$

invariants:
$ENT\_SGN = GRN \Rightarrow IN\_SW \notin OCC$

(abstract_) MOVE_IN
when
$IN\_SW \notin OCC$
...
then
...
end

(concrete_) MOVE_IN
when
$ENT\_SGN = GRN$
...
then
...
$ENT\_SGN := RED$
end

TURN_IN_SW
when
$ENT\_SGN = RED$
then
$IN\_SW :\in PLFS$
end

ALLOW_ENTRY
when
$IN\_SW \notin OCC$
then
$ENT\_SGN := GRN$
end

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Stage 1. To Model the Environment (3/4)

The Signals

variables: $\ldots, ENT\_SGN, EXT\_SGN$

invariants:
$ENT\_SGN = GRN \Rightarrow IN\_SW \notin OCC$

(abstract_) MOVE_IN
when
$IN\_SW \notin OCC$
...
then
...
end

(concrete_) MOVE_IN
when
$ENT\_SGN = GRN$
...
then
...
$ENT\_SGN := RED$
end

TURN_IN_SW
when
$ENT\_SGN = RED$
then
$IN\_SW :\in PLFS$
end

ALLOW_ENTRY
when
$IN\_SW \notin OCC$
then
$ENT\_SGN := GRN$
end

Stage 1. To Model the Environment (3/4)

The Signals

variables: $\ldots, ENT\_SGN, EXT\_SGN$

invariants:
$ENT\_SGN = GRN \Rightarrow IN\_SW \notin OCC$

(abstract_) MOVE_IN
when
$IN\_SW \notin OCC$
$\ldots$
then
$\ldots$
end

(concrete_) MOVE_IN
when
$ENT\_SGN = GRN$
$\ldots$
then
$\ldots$
$ENT\_SGN := RED$
end

TURN_IN_SW
when
$ENT\_SGN = RED$
then
$IN\_SW :\in PLFS$
end

ALLOW_ENTRY
when
$IN\_SW \notin OCC$
then
$ENT\_SGN := GRN$
end

Stage 1. To Model the Environment (3/4)

The Signals

variables: $\ldots, ENT\_SGN, EXT\_SGN$

invariants:
$ENT\_SGN = GRN \Rightarrow IN\_SW \notin OCC$

(abstract_) MOVE_IN
when
$IN\_SW \notin OCC$
...
then
...
end

(concrete_) MOVE_IN
when
$ENT\_SGN = GRN$
...
then
...
$ENT\_SGN := RED$
end

TURN_IN_SW
when
$ENT\_SGN = RED$
then
$IN\_SW :\in PLFS$
end

ALLOW_ENTRY
when
$IN\_SW \notin OCC$
then
$ENT\_SGN := GRN$
end

Stage 1. To Model the Environment (4/4)

The Trains

variables: $\ldots, POS$

invariants:

$POS \in TRAIN \nrightarrow BLOCK$

$\forall t_1, t_2 \cdot t_1 \in \mathrm{dom}(POS) \land t_2 \in \mathrm{dom}(POS) \land t_1 \neq t_2 \Rightarrow POS(t_1) \neq POS(t_2)$

$\mathrm{ran}(POS) = OCC$

(abstract_) MOVE_IN
when
  $\ldots$
  $ENT \in OCC$
then
  $\ldots$
end

(concrete_)MOVE_IN
any t **where**
  $\ldots$
  $t \in \mathrm{dom}(POS)$
  $POS(t) = ENT$
then
  $\ldots$
  $POS(t) := IN\_SW$
end

ARRIVE
any t **where**
  $\ldots$
  $t \notin \mathrm{dom}(POS)$
then
  $\ldots$
  $POS(t) := ENT$
end

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Stage 1. To Model the Environment (4/4)

The Trains

variables: $\ldots, POS$

invariants:

$POS \in TRAIN \nrightarrow BLOCK$

$\forall t_1, t_2 \cdot t_1 \in \mathrm{dom}(POS) \wedge t_2 \in \mathrm{dom}(POS) \;\wedge\; t_1 \neq t_2 \;\Rightarrow\; POS(t_1) \neq POS(t_2)$

$\mathrm{ran}(POS) = OCC$

(abstract_) MOVE_IN
when
$\ldots$
$ENT \in OCC$
then
$\ldots$
end

(concrete_)MOVE_IN
any t **where**
$\ldots$
$t \in \mathrm{dom}(POS)$
$POS(t) = ENT$
then
$\ldots$
$POS(t) := IN\_SW$
end

ARRIVE
any t **where**
$\ldots$
$t \notin \mathrm{dom}(POS)$
then
$\ldots$
$POS(t) := ENT$
end

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Stage 1. To Model the Environment (4/4)

The Trains

variables: $\ldots, POS$

invariants:

$POS \in TRAIN \rightarrowtail\!\!\!\!\!\!+ BLOCK$

$\forall t_1, t_2 \cdot t_1 \in \mathrm{dom}(POS) \land t_2 \in \mathrm{dom}(POS) \;\land\; t_1 \neq t_2 \;\Rightarrow\; POS(t_1) \neq POS(t_2)$

$\mathrm{ran}(POS) = OCC$

(abstract_) MOVE_IN
when
$\ldots$
$ENT \in OCC$
then
$\ldots$
end

(concrete_)MOVE_IN
any t **where**
$\ldots$
$t \in \mathrm{dom}(POS)$
$POS(t) = ENT$
then
$\ldots$
$POS(t) := IN\_SW$
end

ARRIVE
any t **where**
$\ldots$
$t \notin \mathrm{dom}(POS)$
then
$\ldots$
$POS(t) := ENT$
end

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Stage 2. To Model the Actuators (1/2)

The Switch Actuator

variables: $\ldots, act\_in\_sw, act\_in\_sw\_plf$

invariants:

$act\_in\_sw\_plf \in PLFS$

$act\_in\_sw = \mathrm{TRUE} \Rightarrow ENT\_SGN = RED$

TURN_IN_SW

when

$act\_in\_sw = \mathrm{TRUE}$

then

$IN\_SW := act\_in\_sw\_plf$

$act\_in\_sw := \mathrm{FALSE}$

end

ctrl_trigger_in_sw

when

$act\_in\_sw = \mathrm{FALSE}$

$ENT\_SGN = RED$

then

$act\_in\_sw := \mathrm{TRUE}$

$act\_in\_sw\_plf :\in PLFS$

end

Stage 2. To Model the Actuators (2/2)

The Signal Actuator

variables: $\ldots, act\_ent\_sgn$

invariants:

$act\_ent\_sgn = \mathrm{TRUE} \Rightarrow IN\_SW \notin OCC$

$act\_ent\_sgn = \mathrm{FALSE} \lor act\_in\_sw = \mathrm{FALSE}$

ALLOW_ENTRY
when
$act\_ent\_sgn = \mathrm{TRUE}$
then
$ENT\_SGN = GRN$
$act\_ent\_sgn = \mathrm{FALSE}$
end

ctr_chg_ent_sgn
when
$act\_ent\_sgn = \mathrm{FALSE}$
$IN\_SW \notin OCC$
$act\_in\_sw = \mathrm{FALSE}$
then
$act\_ent\_sgn := \mathrm{TRUE}$
end

Stage 3. To Model the Sensors and the Controller (1/2)

The Sensors

invariants:

$snsr\_occc = OCC$

$snsr\_ent\_sgn = ENT\_SGN$

$snsr\_ext\_sgn = EXT\_SGN$

ctrl_trigger_in_sw

when

$act\_in\_sw = \mathrm{FALSE}$

$snsr\_ent\_sgn = RED$

$act\_ent\_sgn = \mathrm{FALSE}$

then

$act\_in\_sw := \mathrm{TRUE}$

$act\_in\_sw\_plf :\in PLFS$

end

ctr_chg_ent_sgn

when

$act\_ent\_sgn = \mathrm{FALSE}$

$IN\_SW \notin snsr\_occ$

$act\_in\_sw = \mathrm{FALSE}$

then

$act\_ent\_sgn := \mathrm{TRUE}$

end

Stage 3. To Model the Sensors and the Controller (1/2)

Controller Variable

variables: $\ldots, ctrl\_in\_sw$

invariants:
$act\_in\_sw = \mathrm{FALSE} \Rightarrow ctrl\_in\_sw = IN\_SW$
$act\_in\_sw = \mathrm{TRUE} \Rightarrow ctrl\_in\_sw = act\_in\_sw\_plf$

(abstract_)ctr_chg_ent_sgn
when
$\ldots$
$IN\_SW \notin snsr\_occ$
$act\_in\_sw = \mathrm{FALSE}$
then
$\ldots$
end

(concrete_)ctr_chg_ent_sgn
when
$\ldots$
$ctrl\_in\_sw \notin snsr\_occ$
$act\_in\_sw = \mathrm{FALSE}$
then
$\ldots$
end

(abstract_)ctrl_trigger_in_sw
when
$\ldots$
then
$\ldots$
$act\_in\_sw\_plf :\in PLFS$
end

(concrete_)ctrl_trigger_in_sw
any p **where**
$\ldots$
$p \in PLFS$
then
$\ldots$
$act\_in\_sw\_plf := p$
$ctrl\_in\_sw := p$
end

Stage 4. To Model some Scheduler

- Simple scheduler: guard strengthening the controller events.

```
ctrl_trigger_in_sw
  any  p  where
    ...
    p ∉ snsr_occ
    p ≠ ctrl_in_sw
    ENTRY ∈ snsr_occ
  then
    ...
  end
```

- More complex scheduler can be modelled via iteration:
  environment - actuators - sensors and controller.

Development Summary

Phase	Model	Proof Obligations
Stage 1	Model 0	0
	Model 1	12
	Model 2	15
	Model 3	23
Stage 2	Model 4	22
	Model 5	29
Stage 3	Model 6	8
	Model 7	7
	Model 8	19
Stage 4	Model 9	0

Summary. Developing Control System

- Start with model of the problem:
  the environment with various constraints.
- Step-by-step introduce:
  - Actuators (output of the controller).
  - Sensors (input of the controller) and the controller.
  - (main difference from Butler's cookbook).
- Schedule the controller appropriately.
- Important features of the approach:
  - Safety properties are introduced early in terms of the environment:
    Safety properties are maintained by refinement.
  - Scheduling details in later phase of the development:
    Separation of concerns between safety properties and schedule.