A Step-wise Development Method with Progress Concerns

Thai Son Hoang
(joint work with Simon Hudon)

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

InfSec Group Seminar, ETH Zurich
26th March 2013

Formal Systems Development using Refinement

- To develop a system **M** satisfying property ϕ, i.e., $\mathbf{M} \models \phi$.
  - **M**: some transition system
  - ϕ: some logical formula
- The main challenge: the complexity of the system.
- Refinement allows the step-by-step design of the system.

Refinement

The UNITY way vs. the Event-B way

- UNITY: Refines the formulae.

$$\overbrace{\phi \Leftarrow \phi_1 \Leftarrow \ldots \Leftarrow \phi_n}^{\text{Refinement}} \; \underbrace{\dashv \; \mathbf{M}}_{\text{Translation}}$$

  - Cons: Hard to understand the choice of refinement.
- Event-B: Refines transition systems.

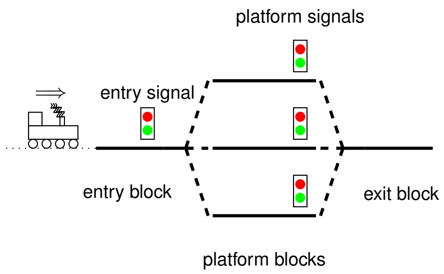
$$\underbrace{\phi \dashv \mathbf{M}_0}_{\text{Verification}} \sqsubseteq \overbrace{\mathbf{M}_1 \ldots \sqsubseteq \mathbf{M}}^{\text{Refinement}}$$

  - Cons: No support for liveness properties.

The Unit-B Modelling Method

- Inspired by UNITY and Event-B.
- Support the reasoning of liveness properties (UNITY).
- Refinement of transition systems (Event-B style).
- Developments using Unit-B are
  guided by both safety and liveness requirements.

Running Example. A Signal Control System



SAF 1 There is at most one train on each block

LIVE 2 Each train in the network eventually leaves

Unit-B Models – Discrete Transition Systems

- States are captured by variables v.
- Transitions are modelled by guarded and scheduled events.

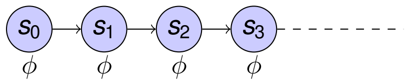
Traces and the Language of Temporal Logic

A trace σ is a (finite or infinite sequence of states)

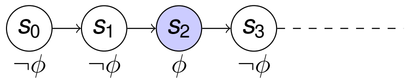
$$\sigma = s_0, s_1, s_2, s_3, \ldots$$

- A (basic) state formula P is any first-order logic formula,
- The basic formulas can be extended by combining the Boolean operators $(\neg, \wedge, \vee, \Rightarrow)$ with temporal operators:

always: $\Box\,\phi$



eventually: $\Diamond\,\phi$



Guarded events

```
e
  any  t  where
    G.t.v
  then
    S.t.v.v'
  end
```

- t: parameters
- $G.t.v$: guard
- $S.t.v.v'$: action

- e.t is enabled when $G.t.v$ holds.
- Execution of e.t: v is updated according to the action $S.t.v.v'$.
- e.t corresponds to a formula act.(e.t).

Scheduled events (1/2)

e
any t **where**
...
during
$C.t.v$
upon
$F.t.v$
then
...
end

- $C.t.v$: coarse-schedule.
- $F.t.v$: fine-schedule.
- Healthiness condition:

$$C.t.v \land F.t.v \Rightarrow G.t.v$$

Liveness (Scheduling) Assumption

If $C.t.v$ holds infinitely long and $F.t.v$ holds infinitely often then eventually e.t is executed.

$$sched.(e.t) = \Box(\Box C \land \Box\Diamond F \Rightarrow \Diamond(F \land act.(e.t)))$$

Schedules vs. Fairness

e $\hat{=}$ **any** t **where** $G.t.v$ **during** $C.t.v$ **upon** $F.t.v$ **then** ... **end**

- Schedules are a generalisation of weak- and strong-fairness.
- Weak-fairness:
  If e is enabled infinitely long then e eventually occurs.
  - Let C be G and F be $\top$.
- Strong-fairness:
  If e is enabled infinitely often then e eventually occurs.
  - Let F be G and C be $\top$.

Scheduled events (2/2)

Conventions

e $\hat{=}$ **any** t **where** ... **during** $C.t.v$ **upon** $F.t.v$ **then** ... **end**

- Unscheduled events (without **during** and **upon**): C is $\bot$
- When only **during** is present (no **upon**), F is $\top$.
- When only **upon** is present (no **during**), C is $\top$.

Execution of Unit-B Models

$$ex.\mathbf{M} = saf.\mathbf{M} \land live.\mathbf{M} \quad (1)$$

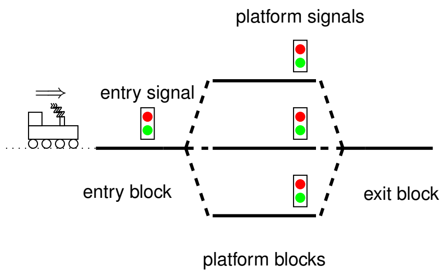
$$saf.\mathbf{M} = init \land \Box\, step.\mathbf{M} \quad (2)$$

$$step.\mathbf{M} = (\exists e.t \in \mathbf{M} \cdot act.(e.t)) \lor \text{SKIP} \quad (3)$$

$$live.\mathbf{M} = \forall e.t \in \mathbf{M} \cdot sched.(e.t) \quad (4)$$

$$sched.(e.t) = \Box(\Box C \land \Box\Diamond F \Rightarrow \Diamond(F \land act.(e.t))) \quad (5)$$

A Signal Control System (Recall)



SAF 1 There is at most one train on each block
LIVE 2 Each train in the network eventually leaves

Refinement strategy: Prioritise LIVE 2 first.

A Signal Control System. The Initial Model

- Focus on trains in the network
- Set *TRN* denotes the set of possible trains.
- Variable *trns* denotes the set of trains in the network.
- Event arrive models a train entering the network.
- Event depart models a train leaving the network.

```
arrive
  any t where
    t ∈ TRN
  then
    trns := trns ∪ {t}
  end
```

```
depart
  any t where
    t ∈ TRN
  during
    t ∈ trns
  then
    trns := trns \ {t}
  end
```


Execution and Properties

Execution and Properties

$\mathbf{M}$ satisfies ϕ if and only if $ex.\mathbf{M} \Rightarrow \phi$.

Safety Properties

- Invariance properties: (in LTL $\Box I$)
  - I holds for every reachable state.
  - Proved using the standard induction technique.
- Unless properties: P **un** Q
  - if P holds at some point then it continues to hold unless Q holds.
  - Prove: If for every event

    e $\widehat{=}$ **any** t **where** $G.t.v$ **during** ... **upon** ... **then** $S.t.v.v'$ **end**

    in **M**, we have

    $$P.v \land \neg Q.v \land G.t.v \land S.t.v.v' \quad \Rightarrow \quad P.v' \lor Q.v' \qquad \text{(UN)}$$

    then **M** satisfies P **un** Q.

Liveness Properties

- Progress properties $P \leadsto Q$.
- In LTL: $\Box(P \Rightarrow \Diamond Q)$
- Some important rules

$$
\begin{array}{rcll}
(P \Rightarrow Q) & \Rightarrow & (P \leadsto Q) & \text{(Implication)} \\
(P \leadsto Q) \land (Q \leadsto R) & \Rightarrow & (P \leadsto R) & \text{(Transitivity)} \\
(P \leadsto Q) & \Leftrightarrow & (P \land \neg Q \leadsto Q) & \text{(Split-Off-Skip)}
\end{array}
$$

A Signal Control System. The Initial Model

Properties

LIVE 2 Each train in the network eventually leaves

properties :
prg0_1 : $t \in trns \leadsto t \notin trns$

Note: Free-variables are universally quantified.

Transient Properties (1/3)

Definition

- Borrowed from UNITY.
- The basic tool for reasoning about progress properties.
- **tr** P states that always P is eventually falsified.
- In LTL: $\Box \Diamond \neg P$.
- Important properties:

$$\textbf{tr}\, P \quad = \quad \top \leadsto \neg P \quad = \quad P \leadsto \neg P$$

Transient Properties (2/3)

Theorem (Implementing tr)

if there exists an event

$e \mathrel{\widehat{=}}$ **any** t **where** $G.t.v$ **during** $C.t.v$ **upon** $F.t.v$ **then** $S.t.v.v'$ **end**

in **M** *such that*

$$
\begin{array}{rr}
\Box(P \Rightarrow C)\,, & \text{(SCH)} \\
C \leadsto F\,, & \text{(PRG)} \\
P.v \land C.t.v \land F.t.v \land G.t.v \land S.t.v.v' \;\Rightarrow\; \neg P.v' & \text{(NEG)}
\end{array}
$$

then **M** *satisfies* **tr** P.

- (SCH) corresponds to an invariance property.
- (PRG) is trivial when F is $\top$.
- (NEG) corresponds to a standard Hoare-triple.

Transient Properties (3/3)

A Sketch Proof

Consider $\textbf{tr}\, P \;=\; P \leadsto \neg P \;=\; \Box(P \Rightarrow \Diamond \neg P)$.

Proof (Sketch).

Assume P holds in some state, we prove $\Diamond \neg P$ by contradiction.

1. Assume $\Box P$.
2. From (SCH), we have $\Box C$,
3. together with (PRG), we have $\Box \Diamond F$.
4. Scheduling assumption ensures that e will eventually occur.
5. (NEG) guarantees that when e occurs, P is falsified.
6. We have a contradiction with the assumption from Step 1

□

A Signal Control System. The Initial Model

Properties

```
depart
  any  t  where
    t ∈ TRN
  during
    t ∈ trns
  then
    trns := trns \ {t}
  end
```

prg0_1 : $t \in trns \leadsto t \notin trns$

- prg0_1 is the same as $\textbf{tr}\; t \in trns$
- (SCH) is trivial.
- No fine-schedule (F is $\top$) hence (PRG) is trivial.
- The event falsifies $t \in trns$ (NEG)

Refinement

- Abstract systems can simulate behaviours of concrete systems.

$$ex.\textbf{cncM} \;\Rightarrow\; ex.\textbf{absM}$$

- Event-based reasoning.

(abs_)e $\widehat{=}$ **any** t **where** G **during** C **upon** F **then** S **end**

(cnc_)f $\widehat{=}$ **any** t **where** H **during** D **upon** E **then** R **end**

- Safety:
  - Guard strengthening: $H \Rightarrow G$
  - Action strengthening: $R \Rightarrow S$
- Liveness:
  - Liveness assumption strengthening.
  - Schedules weakening:

$$(\Box C \;\wedge\; \Box\Diamond F) \;\Rightarrow\; (\Box D \;\wedge\; \Box\Diamond E)$$

Schedules Weakening

Practical Rules

$$(\Box C \;\wedge\; \Box\Diamond F) \;\Rightarrow\; (\Box D \;\wedge\; \Box\Diamond E) \qquad \text{(REF\_LIVE)}$$

- Practical rules:
  - Coarse-schedule following

$$C \wedge F \leadsto D \qquad \text{(C\_FLW)}$$

  - Coarse-schedule stabilising

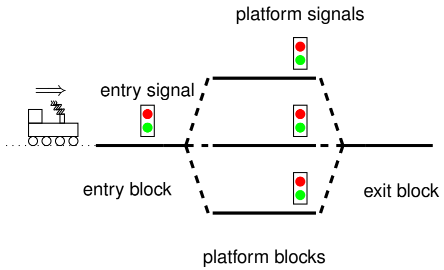
$$D \;\textbf{un}\; C \qquad \text{(C\_STB)}$$

  - Fine-schedule following

$$C \wedge F \leadsto E \qquad \text{(F\_FLW)}$$

A Signal Control System. The First Refinement

The State



- Introduce the network topology: BLK, $Entry$, PLF, $Exit$.
- Variable loc denotes location of trains in the network.

$$\text{inv1\_1} : loc \in trns \rightarrow BLK$$

A Signal Control System. The First Refinement

Refinement of depart

(abs_)depart
any t **where**
$t \in TRN$
during
$t \in trns$
then
$trns := trns \setminus \{t\}$
end

(cnc_)depart
any t **where**
$t \in trns \wedge loc.t = Exit$
during
$t \in trns \wedge loc.t = Exit$
then
$trns := trns \setminus \{t\}$
$loc := \{t\} \lhd loc$
end

- Guard and action strengthening are trivial.
- Coarse-schedule following (amongst others):

$$t \in trns \rightsquigarrow t \in trns \wedge loc.t = Exit \qquad (\text{prg1\_1})$$

A Signal Control System. The First Refinement

Refinement of depart

$t \in trns \rightsquigarrow t \in trns \wedge loc.t = Exit$

$\Leftrightarrow$ Put the negation of RHS in the LHS

$t \in trns \wedge loc.t \neq Exit \rightsquigarrow t \in trns \wedge loc.t = Exit$

$\Leftarrow$ Transitivity

$t \in trns \wedge loc.t \neq Exit \rightsquigarrow t \in trns \wedge loc.t \in PLF \wedge$
$t \in trns \wedge loc.t \in PLF \rightsquigarrow t \in trns \wedge loc.t = Exit$

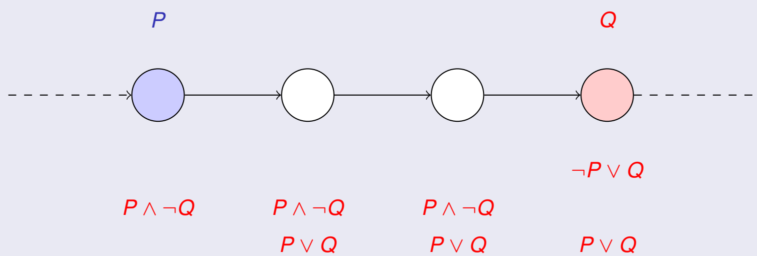
- The 1st condition is implemented by event movein (not shown)
- The 2nd condition is implemented by event moveout
- We need the ensure rule (next slide).

The Ensure Rule

Theorem (The ensure-rule)

For all state predicates p and q,

$$(P \;\mathbf{un}\; Q) \wedge (\mathbf{tr}\; P \wedge \neg Q) \Rightarrow (P \rightsquigarrow Q) \qquad \text{(ENS)}$$



A Signal Control System. The First Refinement

New Event moveout

$t \in trns \land loc.t \in PLF \;\leadsto\; t \in trns \land loc.t = Exit$

$\Leftarrow$ Ensure rule

$t \in trns \land loc.t \in PLF \;\textbf{un}\; t \in trns \land loc.t = Exit \;\land$
$(\textbf{tr}\,(t \in trns \land loc.t \in PLF) \land \neg(t \in trns \land loc.t = Exit))$

$\Leftrightarrow$ Logic

$\ldots \;\land\; (\textbf{tr}\; t \in trns \land loc.t \in PLF)$

```
moveout
  any t where
    t ∈ trns ∧ loc.t ∈ PLF
  during
    t ∈ trns ∧ loc.t ∈ PLF
  then
    loc.t := Exit
  end
```


A Signal Control System. The Second Refinement

The State

SAF 1 There is at most one train on each block

$$\forall t_1, t_2 \;\cdot\; t_1 \in trns \land t_2 \in trns \land loc.t_1 = loc.t_2 \;\Rightarrow\; t_1 = t_2$$

A Signal Control System. The Second Refinement

Refinement of moveout

```
(abs_)moveout
  any t where
    t ∈ trns ∧ loc.t ∈ PLF
  during
    t ∈ trns ∧ loc.t ∈ PLF
  then
    loc.t := Exit
  end
```

```
(cnc_)moveout
  any t where
    t ∈ trns ∧ loc.t ∈ PLF ∧
    Exit ∉ ran .loc
  during
    t ∈ trns ∧ loc.t ∈ PLF
  upon
    Exit ∉ ran .loc
  then
    loc.t := Exit
  end
```

- Neither weak- nor strong-fairness is satisfactory.
  - Weak-fairness requires *Exit* to be free infinitely long.
  - Strong-fairness is too strong assumption.

A Signal Control System. The Third Refinement

The State

- Introduce the signals *sgn*

inv3_1 : $sgn \in \{Entry\} \cup PLF \to COLOR$
inv3_2 : $\forall p \cdot p \in PLF \land sgn.p = GR \Rightarrow Exit \notin \mathrm{ran}\,.loc$
inv3_3 : $\forall p, q \cdot p, q \in PLF \land sgn.p = sgn.q = GR \Rightarrow p = q$

A Signal Control System. The Third Refinement

Refinement of moveout

(abs_)moveout
any t **where**
$t \in trns \land loc.t \in PLF \land$
$Exit \notin \mathrm{ran}\,.loc$
during
$t \in trns \land loc.t \in PLF$
upon
$Exit \notin \mathrm{ran}\,.loc$
then
$loc.t := Exit$
end

(cnc_)moveout
any t **where**
$t \in trns \land loc.t \in PLF \land$
$sgn.(loc.t) = GR$
during
$t \in trns \land loc.train \in PLF \land$
$sgn.(loc.t) = GR$
then
$loc.t := Exit$
$sgn.(loc.t) := RD$
end

Refinement requires to prove:

$$\textbf{tr}\ t \in trns \land loc.t \in PLF \land sgn.(loc.t) = RD\ . \qquad (\text{prg3\_5})$$

A Signal Control System. The Third Refinement

New Controller Event ctrl_platform

ctrl_platform
any p **where**
$p \in PLF \land p \in \mathrm{ran}\,.loc \land Exit \notin \mathrm{ran}\,.loc \land$
$(\forall q \cdot q \in PLF \Rightarrow sgn.q = RD)$
during
$p \in PLF \land p \in \mathrm{ran}\,.loc \land sgn.p = RD$
upon
$Exit \notin \mathrm{ran}(loc) \land (\forall q \cdot q \in PLF \land q \neq p \Rightarrow sgn.q = RD)$
then
$sgn.p := GR$
end

Summary

The Unit-B Modelling Method

- Guarded and scheduled events.
- Reasoning about liveness (progress) properties.
- Refinement preserving safety and liveness properties.
- Developments are guided by safety and liveness requirements.

Summary

Future Work

- Data refinement
- Decomposition / Composition
- Tool support

References I

Simon Hudon.
A Progress Preserving Refinement.
Master Thesis.
Chair of Information Security, ETH Zurich, 2011.

Simon Hudon and Thai Son Hoang.
Systems Design Guided by Progress Concerns.
Accepted for *iFM 2013*.