Modelling Software-based Systems
Lecture 5 The access control problem in Event-B

Telecom Nancy (3A LE)

Dominique Méry

Telecom Nancy,Université de Lorraine

10 avril 2025(12:10 A.M.)
dominique.mery@loria.fr

General Summary

❶ Refinement of models

❷ Summary on Event-B

❸ Case Study The Access Control (J.-R. Abrial)

❹ Conclusion

Summary

Current Summary

- Refinement relates Event-B models
- Problem for starting a refinement-based development
- Problem for finding the best abstract model
- Problem for discharging unproved proof obligations generated for each refinement step
- The Access Control Problem

Current Summary

1. Refinement of models

2. Summary on Event-B

3. Case Study The Access Control (J.-R. Abrial)

4. Conclusion

Events as Relations over Variables Values

- Each variable V has a current value v, a next value v'
- Each event e over variables V is defined by a relation over v and v' denoted $BA(e)(v, v')$.
- An event e has local parameters, variables, guards and actions.
- Events *observe* changes over state variables and changes can be related to code execution or to physical phenomena.

Simple Form of an Event

- An event of the simple form is denoted by :

$$< event\_name > \mathrel{\hat{=}}$$
WHEN
  $< condition >$
THEN
  $< action >$
END

where
- $< event\_name >$ is an identifier
- $< condition >$ is the firing condition of the event
- $< action >$ is a generalized substitution (parallel "assignment")

Non-deterministic Form of an Event

- An event of the non-deterministic form is denoted by :

$$
\begin{array}{l}
< event\_name > \ \widehat{=} \\
\quad \textbf{ANY} \ < variable > \ \textbf{WHERE} \\
\qquad < condition > \\
\quad \textbf{THEN} \\
\qquad < action > \\
\quad \textbf{END}
\end{array}
$$

where
- $< event\_name >$ is an identifier
- $< variable >$ is a (list of) variable(s)
- $< condition >$ is the firing condition of the event
- $< action >$ is a generalized substitution (parallel "assignment")

A generalized substitution can be

- Simple assignment : $x := E$
- Generalized assignment : $x : | P(x, x')$
- Set assignment : $x :\in S$

'

- Parallel composition : $\begin{array}{c} T \\ \cdots \\ U \end{array}$

INVARIANT $\wedge$ GUARD
$\Longrightarrow$
ACTION establishes INVARIANT

- Given an event of the simple form :

```
EVENT e  ≙
   WHEN
      G(x)
   THEN
      x := E(x)
   END
```

and invariant $I(x)$ to be preserved, the statement to prove is :

$$I(x) \ \land \ G(x) \implies I(E(x))$$

- Given an event of the simple form :

```
EVENT e  ≙
  WHEN
    G(x)
  THEN
    x : |P(x, x')
  END
```

and invariant $I(x)$ to be preserved, the statement to prove is :

$$I(x) \ \land \ G(x) \ \land \ P(x, x') \ \implies \ I(x')$$

- Given an event of the simple form :

```
EVENT e  ≙
   WHEN
     G(x)
   THEN
     x :∈ S(x)
   END
```

and invariant $I(x)$ to be preserved, the statement to prove is :

$$I(x) \;\wedge\; G(x) \;\wedge\; x' \in S(x) \;\implies\; I(x')$$

- Given an event of the non-deterministic form :

```
EVENT e  ≙
   ANY v WHERE
      G(x, v)
   THEN
      x := E(x, v)
   END
```

and invariant $I(x)$ to be preserved, the statement to prove is :

$$I(x) \;\land\; G(x, v) \implies I(E(x, v))$$

Refinement Technique (1)

- Abstract models works with variables x, and concrete one with y
- A gluing invariant $J(x, y)$ links both sets of vrbls
- Each abstract event is refined by concrete one (see below)

- Some new events may appear : they refine "skip"
- Concrete events must not block more often than the abstract ones
- The set of new event alone must always block eventually

Correct Refinement Verification (1)

- Given an abstract and a corresponding concrete event

```
EVENT ae  ≙
  WHEN
    G(x)
  THEN
    x := E(x)
  END
```

```
EVENT ce  ≙
  WHEN
    H(y)
  THEN
    y := F(y)
  END
```

and invariants $I(x)$ and $J(x, y)$, the statement to prove is :

$$I(x) \ \wedge \ J(x, y) \wedge H(y) \implies G(x) \wedge J(E(x), F(y))$$

Correct Refinement Verification (1)

- Given an abstract and a corresponding concrete event

```
EVENT ae ≙
  WHEN
    G(x)
  THEN
    x := E(x)
  END
```

```
EVENT ce ≙
  WHEN
    H(y)
  THEN
    y := F(y)
  END
```

and invariants $I(x)$ and $J(x, y)$, the statement to prove is :

$$I(x) \;\land\; J(x, y) \;\land\; H(y) \implies G(x) \;\land\; J(E(x), F(y))$$

- $BA(ae)(x, x') \;\hat{=}\; G(x) \land x' = E(x)$
- $BA(ce)(y, y') \;\hat{=}\; H(y) \land y' = F(y)$

Correct Refinement Verification (2)

- Given an abstract and a corresponding concrete event

$$
\begin{array}{l}
\textbf{EVENT } \text{ae} \ \widehat{=} \\
\quad \textbf{ANY } v \textbf{ WHERE} \\
\qquad G(x, v) \\
\quad \textbf{THEN} \\
\qquad x := E(x, v) \\
\quad \textbf{END}
\end{array}
\qquad
\begin{array}{l}
\textbf{EVENT } \text{ce} \ \widehat{=} \\
\quad \textbf{ANY } w \textbf{ WHERE} \\
\qquad H(y, w) \\
\quad \textbf{THEN} \\
\qquad y := F(y, w) \\
\quad \textbf{END}
\end{array}
$$

$$
\begin{array}{l}
I(x) \ \wedge \ J(x, y) \ \wedge \ H(y, w) \\
\Longrightarrow \\
\exists v \cdot (\, G(x, v) \ \wedge \ J(E(x, v), F(y, w))\,)
\end{array}
$$

- $BA(ae)(x, x') \ \widehat{=} \ \exists v . G(x, v) \wedge x' = E(x)$
- $BA(ce)(y, y') \ \widehat{=} \ \exists w . H(y, w) \wedge y' = F(y)$

Correct Refinement Verification (3)

- Given a NEW event

$$
\begin{array}{l}
\textbf{EVENT } \text{ne} \;\triangleq \\
\quad \textbf{WHEN} \\
\qquad H(y) \\
\quad \textbf{THEN} \\
\qquad y := F(y) \\
\quad \textbf{END}
\end{array}
$$

and invariants $I(x)$ and $J(x, y)$, the statement to prove is :

$$
I(x) \;\wedge\; J(x, y) \;\wedge\; H(y) \;\implies\; J(x, F(y))
$$

- $BA(ne)(y, y') \;\triangleq\; H(y) \wedge y' = E(y)$

Current Summary

1. Refinement of models

2. Summary on Event-B

3. Case Study The Access Control (J.-R. Abrial)

4. Conclusion

- To control accesses into locations.
- People are assigned certain authorizations
- Each person is given a magnetic card
- Doors are "one way" turnstyles
- Each turnstyle is equipped with :
    - a card reader
    - two lights (one green, the other red)

Turnstyle

Access Protocol (after introducing card in reader)

- If access permitted $\left\{\begin{array}{l}\text{- green light turned on} \\ \text{- turnstyle unblocked for 30 sec}\end{array}\right.$

- Passing, or 30 sec elapsed $\left\{\begin{array}{l}\text{- green light turned off} \\ \text{- turnstyle blocked again}\end{array}\right.$

- If access refused $\left\{\begin{array}{l}\text{- red light turned on for 2 sec} \\ \text{- turnstyle stays blocked}\end{array}\right.$

- Sharing between Control and Equipment
- For this : constructing a closed model
- Defining the physical environment
- Possible generalization of problem
- Studying safety questions
- Studying synchronisation questions
- Studying marginal behaviour

P1 : The model concerns people and locations
P2 : A person is authorized to be in some locations
P3 : A person can only be in one location at a time
D1 : Outside is a location where everybody can be
P4 : A person is always in some location
P5 : A person is always authorized to be in his location

Example

Sets

$$\begin{aligned} \text{persons} &= \{\, p1,\ p2,\ p3 \,\} \\ \text{locations} &= \{\, l1,\ l2,\ l3,\ l4 \,\} \end{aligned}$$

Authorizations

p1	l2, l4
p2	l1, l3, l4
p3	l2, l3, l4

Correct scenario

p1	l4
p2	l4
p3	l4

$\rightarrow$

p1	l2
p2	l4
p3	l4

$\rightarrow$

p1	l2
p2	l1
p3	l4

$\rightarrow$

p1	l4
p2	l1
p3	l4

$\rightarrow$

p1	l4
p2	l1
p3	l3

Basic sets : persons P and locations B (prop. P1)
Constant : authorizations A (prop. P2)
A is a binary relation between P and B

$$A \in P \leftrightarrow B$$

Constant : *outside* is a location where everybody is authorized to be (decision D1)

$$outside \in B$$

$$P \times \{outside\} \subseteq A$$

Model (3)

Variable : situations C (prop. P3 and P4)
C is a total function between P and B
A total function is a special case of a binary relation

$$C \in P \to B$$

Invariant : situations compatible with auth. (prop. P5)
The function C is included in the relation A

$$C \subseteq A$$

A magic event which can be observed

- GUARD :
$$\begin{cases} \text{- Given some person } p \text{ and location } l \\ \text{- } p \text{ is authorized to be in } l : p, l \in \mathrm{A} \\ \text{- } p \text{ is not currently in } l : \mathrm{c}(p) \neq l \end{cases}$$

- ACTION : - p jumps into l

```
EVENT observation1   ≜
   ANY p, l WHERE
      p ∈ P   ∧
      l ∈ B   ∧
      p ↦ l ∈ A   ∧
      c(p) ≠ l
   THEN
      c(p) := l
   END
```

Given two relations a and b

Overriding a by b yields a new relation $a \lessdot b$

$$a \lessdot b \quad \widehat{=} \quad (\text{dom}\,(b) \lessdot a) \;\cup\; b$$

Abbreviation

$$f(x) := y \quad \widehat{=} \quad f := f \lessdot \{x \mapsto y\}$$

INVARIANT $\wedge$ GUARD
$\Longrightarrow$
ACTION establishes INVARIANT

$$\text{c} \subseteq \text{A} \ \wedge$$
$$p \in \text{P} \ \wedge$$
$$l \in \text{B} \ \wedge$$
$$p \mapsto l \in \text{A}$$
$$\Longrightarrow$$
$$(\{p\} \lessdot \text{c}) \cup \{p \mapsto l\} \subseteq \text{A}$$

P6 : The geometry define how locations communicate
P7 : A location does not communicate with itself
P8 : Persons move between communicating locations

Constant : communication STRUCTURE (prop. P6 and P7)
STRUCTURE is a binary relation between B
The intersection of STRUCTURE with the identity relation on
B is empty

$$STRUCTURE \in B \leftrightarrow B$$

$$STRUCTURE \cap \mathsf{id}(B) = \varnothing$$

Concrete events do not block more often than abstract ones

$$
\begin{aligned}
&I(x) \ \wedge \ J(x,y) \ \wedge \\
&\text{disjunction of abstract guards} \\
\Longrightarrow \\
&\text{disjunction of concrete guards}
\end{aligned}
$$

New events block eventually (decreasing the same quantity $V(y)$)

$$
I(x) \ \wedge \ J(x,y) \ \wedge \ H(y) \ \wedge \ V(y) = n \implies V(F(y)) < n
$$

Event (prop. P8)
The guard is strengthened
The current location of p and the new location l must communicate

```
EVENT observation1  ≙
  ANY p, l WHERE
    p ∈ P   ∧
    l ∈ B   ∧
    p ↦ l ∈ A   ∧
    c(p) ≠ l
  THEN
    c(p) := l
  END
```

```
EVENT observation2   ≙
  REFINES observation1
  ANY p, l WHERE
    p ∈ P   ∧
    l ∈ B   ∧
    p ↦ l ∈ A   ∧
    c(p) ↦ l ∈ STRUCTURE
  THEN
    c(p) := l
  END
```

Invariant preservation : Success
Guard strengthening : Success

$$\exists\,(p,l)\cdot(\,p \mapsto l \,\in\, \mathrm{A} \,\wedge\, \mathrm{C}(p) \mapsto l \,\in\, \textsc{structure}\,)$$
$$\Rightarrow$$
$$\exists\,(p,l)\cdot(\,p \mapsto l \,\in\, \mathrm{A} \,\wedge\, \mathrm{C}(p) \neq l\,)$$

Deadlockfreeness : Failure

$$\exists\,(p,l)\cdot(\,p \mapsto l \,\in\, \mathrm{A} \,\wedge\, \mathrm{C}(p) \neq l\,)$$
$$\Rightarrow$$
$$\exists\,(p,l)\cdot(\,p \mapsto l \,\in\, \mathrm{A} \,\wedge\, \mathrm{C}(p) \mapsto l \,\in\, \textsc{structure}\,)$$

P9 : No person must remain blocked in a location.

Solution

P10 : Any person authorized to be in a location must also be authorized to go in another location which communicates with the first one.

$$A \subseteq A \, ; \, \text{STRUCTURE}^{-1}$$

$$p \mapsto l \in A \implies \exists m \cdot (\, p \mapsto m \in A \ \wedge \ l \mapsto m \in \text{STRUCTURE} \,)$$

p1	l2	p2	l4
p1	l4	p3	l2
p2	l1	p3	l3
p2	l3	p3	l4

A

l1	l3
l1	l4
l3	l2
l4	l1
l4	l2
l4	l3

STRUCTURE

l1	l4
l2	l3
l2	l4
l3	l1
l3	l4
l4	l1

STRUCTURE^{-1}

p1	l1	p
p1	l3	p
p1	l4	p
p2	l1	p

A; STRUCTU

- Opening a door between l2 and l4
- Authorizing p2 to go to l2
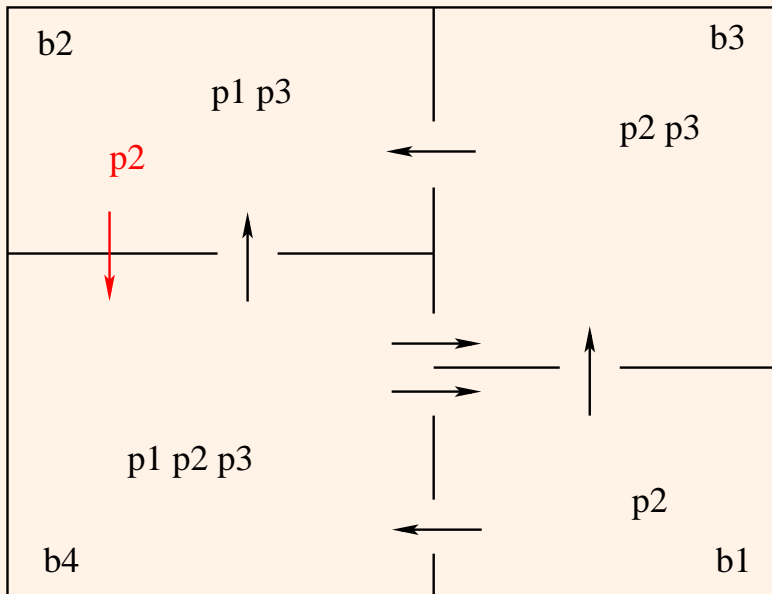
p1	l2	p2	l4
p1	l4	p3	l2
p2	l1	p3	l3
p2	l2	p3	l4
p2	l3		

A

l1	l3
l1	l4
l2	l4
l3	l2
l4	l1
l4	l2
l4	l3

STRUCTURE

l1	l4
l2	l3
l2	l4
l3	l1
l3	l4
l4	l1
l4	l2

$\textsc{structure}^{-1}$

p1	l1	p2
p1	l2	p2
p1	l3	p3
p1	l4	p3
p2	l1	p3
p2	l2	p3

$\text{A}; \textsc{structure}$

Decision

D2 : The system that we are going to construct does not guarantee that people can move "outside".

Constante : $exit$ is a function, included in com, with no cycle

$$exit \in \mathrm{B} - \{outside\} \to \mathrm{B}$$

$$exit \subseteq com$$

$$\forall s \cdot (\, s \subseteq \mathrm{B} \implies (s \subseteq exit^{-1}[s] \implies s = \varnothing)\,)$$

$$\forall x \cdot (x \in s \implies \exists y \cdot (\, y \in s \,\wedge\, (x,y) \in exit\,)\,)$$
$$\implies$$
$$s = \varnothing$$

$exit$ is a tree spanning the graph represented by com

P10' : Every person authorized to be in a location (which is not "outside") must also be authorized to be in another location communicating with the former and leading towards the exit.

$$A \rhd \{outside\} \ \subseteq \ A \ ; \ exit^{-1}$$

$$p \mapsto l \in A \ \wedge$$
$$l \neq outside$$
$$\Longrightarrow$$
$$p \mapsto exit(l) \in A$$

Show that no cycle implies the possibility to prove property by induction and vice-versa

$$\forall s \cdot (\, s \subseteq \text{B} \,\wedge\, s \subseteq exit^{-1}[s] \implies s = \varnothing \,)$$
$$\Leftrightarrow$$
$$\forall t \cdot (\, t \subseteq \text{B} \,\wedge\, outside \in t \,\wedge\, exit^{-1}[t] \subseteq t \implies t = \text{B} \,)$$

$$t \subseteq \text{B}$$
$$outside \in t$$
$$\forall (x, y) \cdot (\, (x \mapsto y) \in exit \,\wedge\, y \in t \implies x \in t \,)$$
$$\implies$$
$$t = \text{B}$$

P11 : Locations communicate via one-way doors.

P12 : A person get through a door only if accepted.

P13 : A door is acceptable by at most one person at a time.

P14 : A person is accepted for at most one door only.

P15 : A person is accepted if at the origin of the door.

P16 : A person is accepted if authorized at destination.

Set : the set DOORS of doors
Constants : The origin ORG and destination DST of a door
(prop. P11)

$$\text{ORG} \in \text{DOORS} \rightarrow \text{B}$$
$$\text{DST} \in \text{DOORS} \rightarrow \text{B}$$
$$\text{STRUCTURE} = (\text{ORG}^{-1} \,;\, \text{DST})$$

Variable : the rel. DAP between persons and doors (prop. P12 to P16)

$$
\begin{aligned}
&\text{DAP} \in P \rightarrowtail\mathrel{\mkern-14mu}\rightarrow \text{DOORS} \\
&(\text{DAP} \,;\, \text{ORG}) \subseteq C \\
&(\text{DAP} \,;\, \text{DST}) \subseteq A
\end{aligned}
$$

P17 : Green light of a door is lit when access is accepted.

P18 : When a person has got through, the door blocks.

P19 : After 30 seconds, the door blocks automatically.

P20 : Red light is lit for 2 sec.when access is refused.

P21 : Red and green lights are not lit simultaneously.

Definition : GREEN is exactly the range of DAP (prop. P17 to P19)

$$GREEN \; \widehat{=} \; ran(DAP)$$

Variable : The set red of red doors (prop. P20)

$$red \subseteq \text{DOORS}$$

Invariant : GREEN and red are incompatible (prop. P21)

$$\text{GREEN} \cap red = \varnothing$$

P22 : Person p is accepted through door d if

      - p is situated within the origin of d

      - p is authorized to move to the dest. of d

      - p is not engaged with another door

$$
\begin{aligned}
&\text{admitted}\,(p,d) \quad \widehat{=} \\
&\quad \mathrm{ORG}(d) = \mathrm{C}(p) \quad \wedge \\
&\quad p \mapsto \mathrm{DST}(d) \in \mathrm{A} \quad \wedge \\
&\quad p \notin \mathsf{dom}\,(dap)
\end{aligned}
$$

Accepting a person p - GUARD :

- Given some person p and door d
- d is neither green nor red
- p is admissible through d

- ACTION :   - make p authorized to pass d

```
EVENT accept  ≙
  ANY p, d WHERE
    p ∈ P  ∧
    d ∈ DOORS  ∧
    d ∉ GREEN ∪ red  ∧
    admitted (p, d)
  THEN
    DAP(p) := d
  END
```

Refusing a person p

- GUARD : $\begin{cases} \text{- Given some person } p \text{ and door } d \\ \text{- } d \text{ is neither green nor red} \\ \text{- } p \text{ is not admissible through } d \end{cases}$

- ACTION : - lit the red light

```
EVENT refuse  ≙
  ANY p, d WHERE
    p ∈ P  ∧
    d ∈ DOORS  ∧
    d ∉ GREEN ∪ red  ∧
    ¬ admitted (p, d)
  THEN
    red := red ∪ {d}
  END
```

Refining Event OBSERVATION2

```
EVENT observation2  ≙
  ANY p, l WHERE
    p ∈ P
    l ∈ B
    p, l ∈ A
    C(p) ↦ l ∈ STRUCTURE
  THEN
    C(p) := l
  END
```

```
EVENT observation3  ≙
  REFINES    observation2
  ANY d WHERE
    d ∈ GREEN
  THEN
    C(DAP⁻¹(d)) := DST(d)
    DAP := DAP ▷ {d}
  END
```

Turning lights off

```
EVENT off_grn  ≙
  ANY d WHERE
    d ∈ GREEN
  THEN
    DAP := DAP ⊳ {d}
  END
```

```
EVENT off_red  ≙
  ANY d WHERE
    d ∈ red
  THEN
    red := red − {d}
  END
```

Synchronization

accept   refuse

↙   ↘   ↓

observation   off_grn   off_red
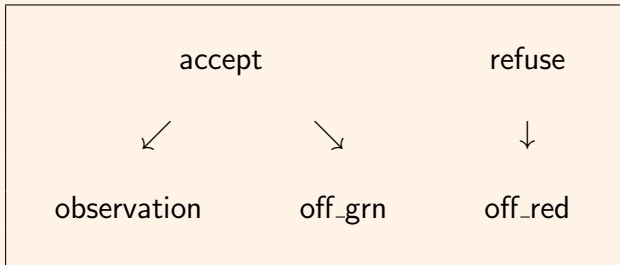
- Event observation is a correct refinement : OK
- Other events refine skip : OK
- Event observation does not deadlock more : OK
- New events do not take control indefinitely : FAILURE

DANGER
- People without the required authorizations
  try indefinitely to enter some locations.
- Other people with the authorization
  always change mind at the last moment.

SOLUTIONS
- Make such practice impossible ? ? ?
- Card Readers can "swallow" a card

D3 : The system we are going to construct will not prevent
people from blocking doors indefinitely :

- either by trying indefinitely to enter places into
  which they are not authorized to enter,
- or by indefinitely abandoning "on the way" their
  intention to enter the places in which they are
  in fact authorized to enter¨.

A decision

D4 : Each card reader is supposed to stay blocked
between :
- the sending of a card to the system
- the reception of an acknowledgement.

The set BLR of blocked Card Readers
The set $mCard$ of messages sent by Card Readers
The set $mAckn$ of acknowledgment messages

$$BLR \subseteq \text{DOORS}$$

$$mCard \in \text{DOORS} \nrightarrow \text{P}$$

$$mAckn \subseteq \text{DOORS}$$

$\mathsf{dom}\,(mCard)$, GREEN, red, $mAckn$ partition BLR

$$\mathsf{dom}\,(mCard)\ \cup\ \text{GREEN}\ \cup\ red\ \cup\ mAckn\ =\ BLR$$

$$\mathsf{dom}\,(mCard)\ \cap\ (\text{GREEN}\ \cup\ red\ \cup\ mAckn)\ =\ \varnothing$$

$$mAckn\ \cap\ (\text{GREEN}\ \cup\ red)\ =\ \varnothing$$

EVENT CARD $\;\hat{=}$
   ANY p, d
   WHERE
     $p \;\in\; \mathrm{P}$
     $d \;\in\; \textsc{doors} - BLR$
   THEN
     $BLR \;:=\; BLR \;\cup\; \{d\}$
     $mCard \;:=\; mCard \;\cup\; \{d \mapsto p\}$
   END

EVENT accept3 $\;\hat{=}$
  ANY p, d
  WHERE
    $p \in$ P
    $d \in$ DOORS
    $d \notin$ GREEN $\cup\ red$
    admitted (p, d)
  THEN
    DAP$(p) := d$
  END

EVENT accept4 $\;\hat{=}$
  REFINES   accept3
  ANY p, d
  WHERE
    $d \mapsto p \in mCard$
    admitted (p, d)
  THEN
    DAP$(p) := d$
    $mCard := mCard - \{d \mapsto p\}$
  END

```
EVENT refuse4  ≙
  REFINES   refuse3
  ANY p, d
  WHERE
    d ↦ p ∈ mCard
    ¬ admitted (p, d)
  THEN
    red := red ∪ {d}
    mCard := mCard − {d ↦ p}
  END
```

EVENT observation4 $\ \widehat{=}$
  REFINES observation3
  ANY d
  WHERE
    $d \in$ GREEN
  THEN
    $\text{C}(\text{DAP}^{-1}(d)) := \text{DST}(d)$
    $\text{DAP} := \text{DAP} \rhd \{d\}$
    $mAckn := mAckn \cup \{d\}$
  END

EVENT off_grn $\;\widehat{=}$
  ANY d **WHERE**
    $d \in$ GREEN
  THEN
    DAP $:=$ DAP $\triangleright \{d\}$
    $mAckn := mAckn \cup \{d\}$
  END

EVENT off_red $\;\widehat{=}$
  ANY d **WHERE**
    $d \in red$
  THEN
    $red := red - \{d\}$
    $mAckn := mAckn \cup \{d\}$
  END

$$
\begin{array}{l}
\textbf{EVENT } \text{ACKN} \ \ \widehat{=} \\
\quad \textbf{ANY } d \textbf{ WHERE} \\
\qquad d \ \in \ mAckn \\
\quad \textbf{THEN} \\
\qquad BLR \ := \ BLR - \{d\} \\
\qquad mAckn \ := \ mAckn - \{d\} \\
\quad \textbf{END}
\end{array}
$$

Synchronization

Decisions

D5 : When a door has been cleared, it blocks itself automatically
     without any intervention from the control system.

D6 : Each door incorporates a local clock for
       - the extinction of the green light after 30 sec.
       - the extinction of the red light after 2 sec.

The set $mAccept$ of acceptance messages (to doors)
The set GRN of physical green doors
The set $mPass$ of passing messages (from doors)
The set $mOff\_grn$ of messages (from doors)

$$mAccept \subseteq \text{DOORS}$$

$$GRN \subseteq \text{DOORS}$$

$$mPass \subseteq \text{DOORS}$$

$$mOff\_grn \subseteq \text{DOORS}$$

$mAccept, GRN, mPass, mOff\_grn$ partition GREEN

$$mAccept \cup GRN \cup mPass \cup mOff\_grn = \text{GREEN}$$

$$mAccept \cap (GRN \cup mPass \cup mOff\_grn) = \varnothing$$

$$GRN \cap (mPass \cup mOff\_grn) = \varnothing$$

$$mPass \cap mOff\_grn = \varnothing$$

The set $mRefuse$ of messages (to doors)
The set RED of phyical red doors
The set $mOff\_red$ of messages (from doors)

$$mRefuse \subseteq \text{DOORS}$$

$$RED \subseteq \text{DOORS}$$

$$mOff\_red \subseteq \text{DOORS}$$

$mRefuse,\ RED,\ mOff\_red$ partition red

$$mRefuse\ \cup\ RED\ \cup\ mOff\_red\ =\ red$$

$$mRefuse\ \cap\ (RED\ \cup\ mOff\_red)\ =\ \varnothing$$

$$RED\ \cap\ mOff\_red\ =\ \varnothing$$

EVENT accept $\ \widehat{=}$
  ANY p, d **WHERE**
    $d \mapsto p \in mCard \ \ \wedge$
    admitted (p, d)
  THEN
    $\mathrm{DAP}(p) \ := \ d$
    $mCard \ := \ mCard - \{d \mapsto p\}$
    $mAccept \ := \ mAccept \ \cup \ \{d\}$
  END

EVENT ACCEPT $\;\widehat{=}$
  ANY d **WHERE**
    $d \in mAccept$
  THEN
    $GRN := GRN \cup \{d\}$
    $mAccept := mAccept - \{d\}$
  END

EVENT PASS $\;\widehat{=}\;$
  ANY d **WHERE**
    $d \in GRN$
  THEN
    $GRN \; := \; GRN - \{d\}$
    $mPass \; := \; mPass \cup \{d\}$
  END

EVENT observation5 $\widehat{=}$
  REFINES observation4 **ANY** d **WHERE**
    $d \in mPass$
  THEN
    $\text{C}(\text{DAP}^{-1}(d)) := \text{DST}(d)$
    $\text{DAP} := \text{DAP} \rhd \{d\}$
    $mAckn := mAckn \cup \{d\}$
    $mPass := mPass - \{d\}$
  END

EVENT OFF_GRN $\;\widehat{=}$
  ANY d **WHERE**
    $d \in GRN$
  THEN
    $GRN := GRN - \{d\}$
    $mOff\_grn := mOff\_grn \cup \{d\}$
  END

EVENT off_grn $\widehat{=}$
  ANY d **WHERE**
    $d \in mOff\_grn$
  THEN
    $\text{DAP} := \text{DAP} \rhd \{d\}$
    $mAckn := mAckn \cup \{d\}$
    $mOff\_grn := mOff\_grn - \{d\}$
  END

EVENT refuse ≙
   ANY p, d **WHERE**
      $d \mapsto p \in mCard$ ∧
      ¬ admitted (p, d)
   THEN
      $red := red \cup \{d\}$
      $mCard := mCard - \{d \mapsto p\}$
      $mRefuse := mRefuse \cup \{d\}$
   END

EVENT REFUSE $\;\widehat{=}$
  ANY d **WHERE**
    $d \in mRefuse$
  THEN
    $RED \;:=\; RED \;\cup\; \{d\}$
    $mRefuse \;:=\; mRefuse - \{d\}$
  END

EVENT OFF_RED $\,\widehat{=}\,$
  ANY d **WHERE**
    $d \in RED$
  THEN
    $RED := RED - \{d\}$
    $mOff\_red := mOff\_red \cup \{d\}$
  END

EVENT off_red $\;\widehat{=}$
  ANY d **WHERE**
    $d \in mOff\_red$
  THEN
    $red := red - \{d\}$
    $mAckn := mAckn \cup \{d\}$
    $mOff\_red := mOff\_red - \{d\}$
  END

Communications

Hardware	Network			Software
CARD	$\rightarrow$	$mCard$	$\rightarrow$	$\begin{cases} \text{accept (1)} \\ \text{refuse (2)} \end{cases}$
ACCEPT	$\leftarrow$	$mAccept$	$\leftarrow$	(1)
PASS	$\rightarrow$	$mPass$	$\rightarrow$	observation (3)
OFF_GRN	$\rightarrow$	$mOff\_grn$	$\rightarrow$	off_grn (3)
REFUSE	$\leftarrow$	$mRefuse$	$\leftarrow$	(2)
OFF_RED	$\rightarrow$	$mOff\_red$	$\rightarrow$	off_red (3)
ACKN	$\leftarrow$	$mAckn$	$\leftarrow$	(3)

Software Data

$$aut \in \mathrm{P} \leftrightarrow \mathrm{B}$$
$$\mathrm{ORG} \in \textsc{doors} \rightarrow \mathrm{B}$$
$$\mathrm{DST} \in \textsc{doors} \rightarrow \mathrm{B}$$
$$\mathrm{A} \subseteq \mathrm{A}; \mathrm{DST}^{-1}; \mathrm{ORG}$$
$$\mathrm{c} \in \mathrm{P} \rightarrow \mathrm{B}$$

$$dap \in \mathrm{P} \rightarrowtail\!\!\!\rightarrow \textsc{doors}$$
$$red \subseteq \textsc{doors}$$

Network data

$$mCard \in \text{DOORS} \nrightarrow \text{P}$$

$$mAckn \subseteq \text{DOORS}$$

$$mAccept \subseteq \text{DOORS}$$

$$mPass \subseteq \text{DOORS}$$

$$mOff\_grn \subseteq \text{DOORS}$$

$$mRefuse \subseteq \text{DOORS}$$

$$mOff\_red \subseteq \text{DOORS}$$

"Physical" Data

$$BLR \subseteq \text{DOORS}$$

$$GRN \subseteq \text{DOORS}$$

$$RED \subseteq \text{DOORS}$$

Software Operations

EVENT test_soft(p, d)

EVENT accept_soft(p, d)

EVENT refuse_soft(d)

EVENT pass_soft(d)

EVENT off_grn_soft(d)

EVENT off_red_soft(d)

Physical Operations

$(p, d) \longleftarrow$ CARD_HARD

ACCEPT_HARD(d)

REFUSE_HARD(d)

$d \longleftarrow$ PASS_HARD

$d \longleftarrow$ OFF_GRN_HARD

$d \longleftarrow$ OFF_RED_HARD

ACKN_HARD(d)

$$(p, d) \longleftarrow \text{read\_card}$$

$$\text{write\_accept}(d)$$

$$\text{write\_refuse}(d)$$

$$d \longleftarrow \text{read\_pass}$$

$$d \longleftarrow \text{read\_off\_grn}$$

$$d \longleftarrow \text{read\_off\_red}$$

$$\text{write\_ackn}(d)$$

$\text{SEND\_CARD}(p, d)$

$d \longleftarrow \text{RCV\_ACCEPT}$

$d \longleftarrow \text{RCV\_REFUSE}$

$\text{SEND\_PASS}(d)$

$\text{SEND\_OFF\_GRN}(d)$

$\text{SEND\_OFF\_RED}(d)$

$d \longleftarrow \text{RCV\_ACKN}$

EVENT CARD $\;\triangleq$
  VAR p, d **IN**
    $(p, d) \longleftarrow$ READ_HARD;
    SEND_CARD(p, d)
  END

EVENT accept_refuse $\;\triangleq$
  VAR p, d, b **IN**
    $(p, d) \longleftarrow$ read_card;
    $b \longleftarrow$ **EVENT** test_soft(p, d);
    IF $b =$ true **THEN** **EVENT** accept_soft(p, d); write_accept(d)
    ELSE **EVENT** refuse_soft(d); write_refuse(d) **END**
  END

EVENT ACCEPT $\;\triangleq$
  VAR d **IN**
    $d \longleftarrow$ RCV_ACCEPT;
    ACCEPT_HARD(d)
  END

EVENT REFUSE $\;\triangleq$
  VAR d **IN**
    d $\longleftarrow$ RCV_REFUSE;
    REFUSE_HARD(q)
  END

EVENT PASS ≙
  VAR d **IN**
    $d \longleftarrow$ PASS_HARD;
    SEND_PASS(d)
  END

EVENT OFF_GRN ≙
  VAR d **IN**
    $d \longleftarrow$ OFF_GRN_HARD;
    SEND_OFF_GRN(d)
  END

EVENT OFF_RED ≙
  VAR d **IN**
    $d \longleftarrow$ OFF_RED_HARD;
    SEND_OFF_RED(d)
  END

EVENT observation ≙
  VAR d **IN**
    $d \longleftarrow$ read_pass;
    EVENT pass_soft(d);
    write_ackn(d)
  END

EVENT off_grn ≙
  VAR d **IN**
    $d \longleftarrow$ read_off_grn;
    EVENT off_grn_soft(d);
    write_ackn(d)
  END

EVENT off_red ≙
  VAR d **IN**
    $d \longleftarrow$ read_off_red;
    EVENT off_red_soft(d);
    write_ackn(d)
  END

EVENT ACKN ≙
  VAR d **IN**
    d $\longleftarrow$ RCV_ACKN;
    ACKN_HARD(d)
  END

22 Properties et 6 "System" Decisions - One Problem
Generalization
- Access between locations
- One Negative Choice :
- Possible Card Readers Obstructions
- Three Physical Decisions
- Automatic Blocking of Doors
- Automatic Blocking of Card Readers
- Setting up of Clocks on Doors
- The overall development required 183 proofs
- 147 automatic (80%)
- 36 interactive

Current Summary

1. Refinement of models

2. Summary on Event-B

3. Case Study The Access Control (J.-R. Abrial)

4. Conclusion

Conclusion

- Identify an abstract model
- Identify constants and states
- Identify components
- Plan the refinement
- Start as long as the model is not well defined !

- A is a variable which can be modified by events mdeolling the administration of the access control model :
  - ▶ adding authorizations to a set of persons
  - ▶ removing or deleting authorizations of a set of persons
- Generalizing to other problems :
  - ▶ a set of users U has access to a set of resources R.
  - ▶ a set of rooms R is managed by a set of keycards K.
  - ▶ a set of users U has access to a set of services S.