XVI. A Location Access Controller (October 2008)

The purpose of this chapter is to study yet another example dealing with a complete system like the one we studied in chapter 2, where we controlled cars on a bridge, and in chapter 3, where we studied a mechanical press controller. The system we study now is a little more complicated than the previous ones. In particular, the mathematical data structure we are going to use is more advanced. Our intention is also to show that during the reasoning on the model we shall discover a number of important missing points in the requirement document.

1 Requirement Document

We shall construct a system which will be able to control the access of certain people to different locations of a "workplace" : for example, a university campus, an industrial site, a military compound, a shopping mall, etc.

The system concerns people and locations	FUN-1

The control takes place on the basis of the authorization that each concerned person is supposed to possess. This authorization should allow him, controlled by the system, to penetrate into certain locations, and not into others. For example, a certain person p_1 is authorized to enter location l_1 and not location l_2; however, another person p_2 is allowed to enter both locations. These authorizations are given on a "permanent" basis: in other words, they will not change during a normal functioning of the system.

People are permanently assigned the authorization to access certain locations	FUN-2

A person which is in a location must be authorized to be there	FUN-3

When someone is inside a location, his eventual exit must also be controlled by the system, so as to be able to know at any moment who is inside a given location. Each involved person receives a magnetic card with a unique identifying sign, which is engraved on the card itself.

Each person receives a personal magnetic card	EQP-1

Card readers are installed at each entrance and at each exit of concerned locations. Near to each reader, two control lights can be found: a red one and a green one. Each one of these lights can be on or off.

Each entrance and exit of a location is equipped with a card reader	EQP-2

Each card reader has two lights: one green light and one red light	EQP-3

Each light can be “on” or “off”.	EQP-4

The transfer of people from one location to another takes place thanks to “turnstiles” which are normally blocked: nobody can get through them without being controlled by the system, any person getting through is detected by a sensor.

Locations communicate via one-way turstiles	EQP-5

Each turnstile is only affected to a single task, either entry or exit, there are no “two-way” turnstiles. Turnstiles and card readers are illustrated on figure 1.

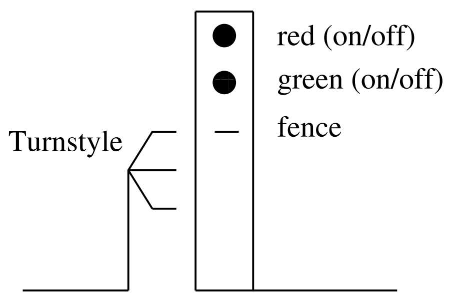

Fig. 1. Turnstile ansd Card Reader

A sensor detects the passage of a person through a turnstile	EQP-6

Turnstiles are normally blocked	FUN-4

The entry or the exit of a location follows a systematic procedure composed of a suite of events. A person wishing to enter or exit a location puts his card into the card reader on the appropriate turnstile. One is then faced with the following alternatives:

– If the person is authorized to pass through the turnstile, the green control light is lit and the turnstile is opened during 30 seconds. We are now faced with the following situation:

    - As soon as the individual gets through the turnstile within the 30 seconds limit, the green control light goes out immediately and the turnstile blocks.
    - If, however, 30 seconds go by without anybody going through the turnstile, the control light goes out and the turnstile is also blocked.
- If the person is not authorized to pass through the turnstile, the red control light goes on for two seconds and, of course, the turnstile remains blocked.

A person willing to pass through a turnsile puts its card in the fence of the card reader	FUN-5

If the person is accepted, the green light is lit and the turnstile is unblocked for at most 30 seconds.	FUN-6

If the person is not accepted, the red light is lit for 2 seconds and the turnstile remains blocked	FUN-7

As soon as an accepted person has gone through an unblocked turnstile, the green light is turned off and the turnstile is blocked again	FUN-8

If nobody goes through an unblocked turnstile during the 30 seconds period, the green light is turned off and the turnstile is blocked again	FUN-9

2 Discussion

The above informal presentation of the system does not pretend to be complete, or to have raised all the technical options. Indeed, it is merely the minimal starting point for the realization of the future control software, but clearly there remains many unknowns concerning, among others, the hardware and its links with the software. We give precisions hereafter as to the role of the study which we shall undertake during the construction of the formal model.

2.1 Sharing out of the Control

An important question which should be asked at the outset about such a system concerns the distribution, which is more or less important, of control between the software and the diverse peripherals (turnstiles, readers).

For example, there can be a computer at each turnstile: in this case, the control is entirely decentralized. Inversely, a single computer can entirely centralize the control. Of course, there are intermediate situations in which each turnstile has a certain form of autonomy: an example consists in equipping each turnstile with a clock which conditions part of its behavior.

2.2 Construction of a Closed Model

In any case, technical argumentation which can lead to such and such a decision can only be realized *by analyzing the system as a whole*. It is to be noted that this technical argumentation must also be nourished by information concerning the equipments available on the market (it can also be decided to realize new equipment).

For this we shall therefore build a *closed model* of our future system and *prove* that its *characteristic properties* (which will have to be precisely explained) are in fact assured.

2.3 Behavior of the Equipments

An important result of this study concerns the behavioral specification of different equipments. So as to conduct this study correctly, we will be obliged to introduce into the model a certain amount of supposition concerning these equipments. For example, does the turnstile block of its own accord after the pass of a single person, or does it block only after the reception of an order from the software? Of course, the chosen option conditions the organization of the software. This option makes up a hypothesis under which the software can function correctly.

Thus, it will be necessary to make a certain number of choices concerning the behavior of the equipment. These choices will result, among other things, in the definition of a receipt procedure aimed at verifying that the equipments which have been installed have in fact the expected qualities. If this were not the case, it is clear that the union of the software and the hardware would have no chance of working correctly as had been demonstrated on the model. This demonstration will therefore have been useless.

2.4 Tackling Safety Questions

An important question which is not tackled at all in the requirement document concerns the safety of people implicated in this system. We would like the model to inform us on this point, or at least that it asks a certain number of pertinent questions. For example, can people be blocked for ever in a location? How can we guarantee the contrary?

2.5 Synchronization Problems

On a more technical level, the informal presentation says nothing about the details of the timing of the transfer operation. For example, what time-lag is there between the lighting up of the green control light, the acceptance of the turnstile and the start of the 30 second countdown? Could the green light go on while the turnstile is still blocked, or could it go out while the turnstile is still accepted?

The question is not so much to find out if the previous behavior is good or bad, it is rather to recognize the existence of a certain behavior and to know in advance if it can occur (even in a fugitive form) in the final system which will be installed.

2.6 Functioning at the Limits

Another question which is not treated in the requirement document is what happened when the system functions “at the limits”. For example, what is the reaction of the system when a card is introduced into the reader whilst the green or red lights are still lit (that is to say, before the preceding *action* is finished)? More generally, we would like to be able to understand and predict how the system reacts when faced with “hostile” behavior of certain users. In this sort of system, one must not count upon hypotheses which rely too much upon the “good” behavior of users. Some users certainly do behave in a “strange” way.

3 Initial Model of the System

We are now going to proceed to the initial formal description of our system by constructing a first, very abstract, model in which the separation between the software and the hardware is not the current issue.

In this first model we introduce the set P of person and the set L of locations as carrier sets. We remind the reader that a carrier set has the implicit unique property of being non-empty. We introduce a constant aut denoting the authorization permanently given to persons: it is a binary relation built on the sets P and L as stated in **axm0_1**. We also introduce a special location, out, denoting the outside. We ensure in **axm0_3** that everyone is authorized to be in location out!

Finally, we introduce a variable, sit, denoting the location where each person is. Note that this is a total function as stated in **inv0_1**: a person cannot be in two locations at a time and a person is always somewhere (thanks to out). Finally, the main property of our system is stated in invariant **inv0_2**: every person, which in a certain location, is authorized to be there. This invariant formalizes requirement FUN_3. Here is our initial formal state:

sets: P
L

constants: aut
out

axm0_1: $aut \in P \leftrightarrow L$

axm0_2: $out \in L$

axm0_3: $P \times \{out\} \subseteq aut$

variables: sit

inv0_1: $sit \in P \rightarrow L$

inv0_2: $sit \subseteq aut$

Initially, everyone is outside as indicated in event init. We have a unique normal event, pass, corresponding to a person going from one location to another different one.

```
init
    $sit := P \times \{out\}$
```

```
pass
  **any** $p, l$ **where**
      $p \mapsto l \in aut$
      $sit(p) \neq l$
  **then**
      $sit(p) := l$
  **end**
```

To illustrate this, let us suppose that we have four locations l1, l2, l3, and l4 and three persons p1, p2, and p3 together with the following authorizations:

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

The following situations can therefore represent a satisfying evolution of the system since, as can be noted, invariant **inv0_1** and **inv0_2** are respected:

p1	l4
p2	l4
p3	l4

Situation 1

p1	l2
p2	l4
p3	l4

Situation 2

p1	l2
p2	l1
p3	l4

Situation 3

p1	l4
p2	l1
p3	l4

Situation 4

p1	l4
p2	l1
p3	l3

Situation 5

It is easy to *prove* that the invariants are well preserved by the above transitions when they happen. It is to be remarked that the event pass is very abstract; we do not know by which process the person p enters the location l. Neither do we know if the location in which the person p is, communicates with location l in which he wishes to go. In fact, we cannot express this property because we have not yet formalized the "geometry" of the locations.

The only important element of the present model is the expression of fundamental rules of the system in invariant **inv0_1** and **inv0_2**, and the proof that the unique observable and interesting event at this level (pass) maintains these conditions. We are already sure that the future models will respect these conditions if, of course, we can *prove* that they constitute *correct refinements* of the present model.

4 First Refinement

4.1 State and Event

We are now going to proceed to our first refinement which will consist in introducing into the model the notion of possible direct communication between two locations. For this, we introduce a new constant, com, denoting the direct communication between two locations. This is a binary relation built on the set L. A location does not "communicate" with itself as stated in **axm1_2**.

constants: $\dots$
com

axm1_1: $com \in L \leftrightarrow L$

axm1_2: $com \cap \mathrm{id} = \varnothing$

The initialization event does not change, and the event pass is refined in a straightforward way: we stipulate that a person can move to another location l if it has the authorization to be in l (as already stated in the abstraction) and also if location l communicates with the location where p is now, that is $sit(p)$.

```
init
    $sit := P \times \{out\}$
```

```
pass
  any $p, l$ where
    $p \mapsto l \in aut$
    $sit(p) \mapsto l \in com$
  then
    $sit(p) := l$
  end
```

This event is quite simply a refinement of the previous version because the action is identical in both cases ($sit(p) := l$) and the guard of the second is stronger than that of the first as we have:

$$sit(p) \mapsto l \in com \Rightarrow sit(p) \neq l$$

since a location can't communicate with itself according to **axm1_2**.

4.2 Deadlock Freeness

It must now be proved that, in the absence of new events in this refinement, the concrete event pass does not happen less often than its abstract homologue (we remind you that this is a necessary condition of the refinement). In fact, we are obviously faced with a difficulty here: it is not possible to prove that the refined event pass does not happen less often than its more abstract homologue. For demonstrating this, we would have to prove that the guard of the abstract event implies that of the concrete event, that is:

$$\exists p, l \cdot p \mapsto l \in aut \;\; \wedge \;\; sit(p) \neq l \;\; \vdash \;\; \exists p, l \cdot p \mapsto l \in aut \;\; \wedge \;\; sit(p) \mapsto l \in com$$

It is clear that this condition can not be verified in general. A counter-example is easy to exhibit. Suppose we only have one person, p, in our system, and suppose that p is in a location $sit(p)$ where one can enter (p did it). Now if this location has no exit then p cannot leave it to another location l, although it could do it in the abstraction because the constraints about communication between locations did not exist.

The failure to prove the above condition indicates that *there are possibilities that some persons could stay permanently blocked in locations*. And to be more precise, this could be the case even if this possibility does not exist in the abstraction, that is to say even if the authorizations are well defined to begin with so that it cannot happen. In fact, the geometry of communication between locations clearly adds an additional constraint limiting the way people can move.

Indeed, if a person is in a location l and has no authorization allowing him to be in any of the locations which communicate with l, then that person is blocked in l. The impossibility to make the above proof has brought up a safety problem, which can be set out in the form of a safety requirement which the system must satisfy:

No persons must remain blocked in a location	SAF-1

Notice that, what is stated here is stronger than what would have been needed, strictly speaking, to allow us to perform the proof that failed. That weaker statement would have been the following: "the geometry of locations does not introduce additional possibilities of blockage beyond those already present without this constraint". As a matter of fact, the mathematics of our approach has revealed a problem that gives us the idea of a wider safety question that was totally ignored in the requirement document.

4.3 A First Solution

So we must find a sufficient constraint so that this requirement SAF-1 is practically satisfied. The previous proof obligation, namely

$$\exists p, l \cdot p \mapsto l \in aut \;\; \wedge \;\; sit(p) \neq l \;\; \vdash \;\; \exists p, l \cdot p \mapsto l \in aut \;\; \wedge \;\; sit(p) \mapsto l \in com$$

will serve as a model for this constraint. It is sufficient to prove the consequent of this implication:

$$\exists\, q, m \cdot q \mapsto m \,\in\, aut \;\;\wedge\;\; sit(q) \mapsto m \,\in\, com$$

which can be re-written as follows:

$$(aut \;\cap\; (sit; com)) \neq \varnothing$$

or, in an equivalent way

$$((aut; com^{-1}) \;\cap\; sit) \neq \varnothing$$

So as to prove this condition, it is sufficient to prove the following (since the condition $P \neq \varnothing$ implies that the total function sit is not empty):

$$sit \;\subseteq\; (aut; com^{-1})$$

What does this condition say? If we develop it, we obtain:

$$\forall p \cdot \exists l \cdot (\, p \mapsto l \in aut \;\;\wedge\;\; sit(p) \mapsto l \in com\,)$$

In other words, the location $sit(p)$, in which each person p is situated at any moment, is in communication with at least one other location l in which p is authorized to go: the person p can therefore go out of the the location $sit(p)$, in which he is, via l.

This condition could be imposed as a new invariant of the system: however, it would be necessary to reinforce the guard for the pass event so as to admit in any one location only the persons authorized to enter it (this is already the case), and who would also be authorized to exit. If faced with an interdiction, the authorization to enter such a location, which the person concerned would hold, would not be of much use since access would be refused for him anyway. So, it would be preferable to have a (sufficient) condition *which would be independent of the situation of people*. Indeed, this is possible, because we already have the invariant property $sit \;\subseteq\; aut$. So as to prove the condition $sit \;\subseteq\; (aut; com^{-1})$ above, it is sufficient to prove the following, by transitivity of the inclusion:

$$\boxed{aut \;\subseteq\; aut; com^{-1}}$$

This condition makes up a further invariant, which can be translated as follows:

$$\forall\, p, l \cdot p \mapsto l \in aut \;\;\Rightarrow\;\; (\,\exists\, m \cdot p \mapsto m \in aut \;\;\wedge\;\; l \mapsto m \in com\,)$$

The interpretation of this invariant is quite instructive : it can be remarked that whenever the pair $p \mapsto l$ belongs to aut (the person p could therefore be in the location l since he is authorized to be there), there is a location m such as $p \;\mapsto\; m$ belongs to aut (the same person p is therefore authorized to enter the location m) and, moreover, such that the pair $l \;\mapsto\; m$ belongs to com (so the two locations l and m do communicate). When all is said and done, the person p, who could be in the location l, would not remain blocked indefinitely since he is also authorized to enter the location m which communicates with l. So this person can leave l via m. As can be seen, requirement SAF-1 has now been satisfied thanks to a stronger requirement whose expression has been determined (calculated) before being expressed in English as follows:

Any person authorized to be in a location must also be authorized to go in another location which communicates with the first one.	SAF-2

4.4 Second Solution

It is to be noted that the previous solution is not satisfactory. It should be widened to guarantee that any person in a location, can not only go out of it but, more generally, can also get outside. For this we extend our constant by introducing a function, $exit$, connecting locations to locations and defined at every location except out (this is property **axm1_3** below). More precisely, exit defines a tree structure. As a consequence, we can copy the tree axioms as defined in section 7.6 of chapter 9. Moreover, $exit$ must be compatible with com (**axm1_5**). Finally, we must state that people must be authorized to follow the exit sign (**axm1_6**). All this leads to the following axioms:

constants: $\ldots$
$exit$

axm1_3 : $exit \in L \setminus \{out\} \to L$

axm1_4 : $\forall s \cdot s \subseteq exit^{-1}[s] \;\Rightarrow\; s = \varnothing$

axm1_5 : $exit \subseteq com$

axm1_6 : $aut \rhd \{out\} \;\subseteq\; aut \,;exit^{-1}$

The last property could be promoted to a requirement replacing the one we defined in the previous section

Any 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 outside.	SAF-3

4.5 Revisiting Deadlock Freeness

In previous section, we have proved that people which are in any location (except "outside") can always go outside. But we have not proved that we have no deadlock for people which are outside! We must prove that people can enter into the building. This is formalized by means of the following additional property:

axm1_7 : $\forall p \cdot p \in P \;\Rightarrow\; (\,\exists l \cdot p \mapsto l \subseteq aut \;\wedge\; out \mapsto l \in com\,)$

5 Second Refinement

5.1 State and Events

During this second refinement, we are going to introduce one-way doors to communicate from one location to another. The formalization goes through the introduction of a new carrier set D which makes models of the doors. Each door is associated with a location of origin, represented by the total function org (property **axm2_1**) and a destination location, represented by the total function dst (property **axm2_2**). For all these doors, the locations of origin and destination represent exactly the pairs of locations implied in the relation com introduced during the previous refinement (property **axm2_3**). This is formalized as follows:

sets:	...
	D

constants:	...
	org
	dst

axm2_1 :	$org \in D \to L$
axm2_2 :	$dst \in D \to L$
axm2_3 :	$com = (org^{-1} \, ; dst)$

We introduce three new variables in this refinement, namely dap, grn, and red. Variable dap is a partial function from the set P of persons to the set D of doors.

variables:	...
	dap

inv2_1 :	$dap \in P \rightarrowtail\!\!\!\!\!\!+\; D$
inv2_2 :	$(dap \, ; org) \subseteq sit$
inv2_3 :	$(dap \, ; dst) \subseteq aut$

It corresponds to the temporary connection that exists between a person p willing to go through a door d, but which has not passed yet through the door d. This connection exists between the moment where that person is "accepted" by the door (new event accept) and the subsequent moment where either that person passes through the door (event pass) or the 30 seconds delay is over (new event off_grn). Variable dap is a function (invariant **inv2_1**) because we do not want a person to be involved with more that one door at a time (since otherwise some additional persons could be admitted into locations without card). And it is also an injective function (invariant **inv2_1**) because we do not want to have a door involved with more than one person at a time (since otherwise people could be confused with the meaning of the green and red lights). Moreover, the origin of the door involved in this relationship correspond to the actual situation of the person (invariant **inv2_2**) and the destination of the door is a location where the person is authorized to be (invariant **inv2_3**):

The other two variables grn and red denotes the subsets of the doors where the green or red light is lit (invariants **inv2_4** and **inv2_5**). Note that the set of green doors exactly corresponds to the range of the variables dap (invariant **inv2_6**). Also notice that both lights cannot be lit simultaneously (invariant **inv2_7**):

variables:	...
	grn
	red

inv2_4 :	$grn \subseteq D$
inv2_5 :	$red \subseteq D$
inv2_6 :	$grn = \mathrm{ran}(dap)$
inv2_7 :	$grn \cap red = \varnothing$

We have two new events, accept and refuse, defined below:

```
accept
  any $p, d$ where
    $p \in P$
    $d \in D$
    $d \notin grn \cup red$
    $sit(p) = org(d)$
    $p \mapsto dst(d) \in aut$
    $p \notin \mathsf{dom}\,(dap)$
  then
    $dap(p) := d$
    $grn := grn \cup \{d\}$
  end
```

```
refuse
  any $p, d$ where
    $p \in P$
    $d \in D$
    $d \notin grn \cup red$
  $\neg\,($ $sit(p) = org(d)$
    $p \mapsto dst(d) \in aut$
    $p \notin \mathsf{dom}\,(dap)\,)$
  then
    $red := red \cup \{d\}$
  end
```

Both events involve a person p and a door d where red and green lights are both off. Event pass has three additional guards which make the door d possibly accepting the person p: (1) person p must be situated at the origin of d, (2) person p is authorized to go at the destination of d, and finally (3) person p is not already involved with a door. Event refuse has an additional guard which is the negation of the conjunction of the three previous guards.

Next are the definitions of two new events, off_grn and off_red, and also the refinement of the abstract event pass:

```
off_grn
  any $d$ where
    $d \in grn$
  then
    $dap := dap \rhd \{d\}$
    $grn := grn \setminus \{d\}$
  end
```

```
off_red
  any $d$ where
    $d \in red$
  then
    $red := red \setminus \{d\}$
  end
```

```
pass
  any $d$ where
    $d \in grn$
  with
    $p = dap^{-1}(d)$
    $l = dst(d)$
  then
    $sit(dap^{-1}(d)) := dst(d)$
    $dap := dap \rhd \{d\}$
    $grn := grn \setminus \{d\}$
  end
```

Event off_grn corresponds to the green light being put off when the 30 seconds delay has passed. Event off_red corresponds to the red light being put off when the 2 seconds delay has passed. Event pass corresponds to the person p associated with a green door d passing that door. The person p in question is $dap^{-1}(d)$. The green light of d is put off and the association between p and d is removed.

5.2 Synchronization

As illustrated in the following diagram, the synchronization between the different events is quite weak for the moment.

$$\begin{tikzcd}
& \text{accept} \arrow[dl] \arrow[dr] & & \text{refuse} \arrow[d] \\
\text{pass} & & \text{off\_grn} & \text{off\_red}
\end{tikzcd}$$

5.3 Proofs

It is easy to prove that the new version of the event pass refines its abstraction. It is also easy to prove that the new events all refine the event which does nothing, skip. Two things remain to be proved: (1) that the event pass does not happen less often than its abstraction, taking into account, of course, the new events, and (2) that the new events cannot take control indefinitely, thus preventing the event pass from taking place.

In fact, the proof of (1) is relatively easy, however that of (2) is quite simply *impossible*. We have here a new difficulty which we will have to investigate and probably correct by additional requirements.

5.4 Risk of Permanent Obstruction of Card Readers

Taking into account the initial informal presentation, we now know that the event pass envisaged above is not the only "basic" event which can be observed. We know that the entrance of a person can be refused and also that a person who wishes to enter a location can change his mind just before passing: in this last case the door is automatically blocked again after 30 seconds. This corresponds to the events refuse and off_grn envisaged above.

However, during their introduction, it must be proved that these two events cannot prevent indefinitely the event pass from happening (that is to say that their guards are not indefinitely true at the same time as those of the event pass) which, in theory rather than in practice, is not impossible.

Indeed, we can imagine observing strange behavior of the system where nobody could ever enter the locations, either because people without the necessary authorizations keep trying to get in, or because other people, who are authorized put their cards in the reader but change their minds at the last minute. So as to prove that this behavior is not indefinitely possible, we must propose something to prevent them.

5.5 Propositions for Preventing Permanent Obstruction

A first way of proceeding would be to formalize a mechanism whereby the "system" (at large) would force, in one way or another, people who are not allowed to access a location not to keep on *indefinitely* trying to get in (meeting indefinitely with refuse). In the same way, should the system force, in one way or another, those who have the right, not to give up at the last minute (thus provoking *indefinitely* a new blockage). This type of drastic behavior, in all evidence, is not part of the specification of the system we are analyzing.

A second way of proceeding, more gentle but also more efficient, would consist in eliminating from the system all the persons who tend to behave in this way too frequently. They would simply no longer be allowed to enter any location at all. Their authorization to enter any location at all would be taken away. These persons would therefore be confined to the location in which they find themselves, for example "outside". This is very easy to formalize and then to realize: moreover, this is the situation to be found in most smart card systems. For example, after three successive unfruitful tries with a cash dispenser, the card is "snatched", which is a very efficient way of preventing the person in question from blocking indefinitely the access to the dispenser. Note however that such a drastic solution has its drawback in our case: the person whose card has been removed simply cannot leave the location where this has happened, causing yet another safety problem.

5.6 Final Decision

At the issue of this discussion we decide not to envisage this last possibility which would make the card readers too complicated (too expensive) and would introduce an additional safety problem. In other words, we accept, for financial (and safety) reasons, a risk of indefinite obstruction, which, despite everything, will probably never happen in reality. In other words, the system we are going to construct will not prevent people from blocking doors indefinitely either by trying indefinitely to enter locations into which they are not authorized to enter, or by abandoning "on the way" their intention to enter the locations in which they are in fact authorized to enter.

Clearly, the system we are going to construct is thus not totally correct according to our theoretical criteria. We accept this, but, and this is very important, we have taken great care to make it known, and to point it out explicitly by a clearly expressed decision.

6 Third Refinement

6.1 Introducing the Card Readers

We now introduce the card readers into the model. This is the first time we will have been taking into account such a material element. This device can be characterized by: (i) the capture of information which is read on the card introduced by the user and (ii) the expedition of this information towards the controlling computer by the means of a networked message. Moreover, when a card is read, it can be supposed that the reader is physically blocked (the slot is obstructed) until it receives an acknowledgement message coming from the control system.

All this corresponds to the following behavioral decision for card readers: Each card reader is supposed to stay physically blocked (slot obstructed) between the moment when the contents of a card is sent to the system and the reception by this reader of the corresponding acknowledgement. This comes when the pass protocol has been entirely completed (successfully or not).

By this decision, we are making sure that no-one will be able to introduce a card into a reader at random. It is to be noted that we must pay a certain price for this, that of the installation of readers with obstructable slots (they do not all belong to this category).

6.2 Assumptions Concerning the Communication Network

We will only make minimal assumptions concerning the forwarding of messages through the network. For example, we suppose that the network does not guarantee that the messages be received in the order in which they have been sent. However, it can be supposed that the messages sent along the network are neither lost, nor modified, nor duplicated. Of course, we could take into account these particular constraints but then the model would be more complicated. In any case, such constraints would only be introduced during ulterior refinements.

6.3 Variables and Invariant

We will identify in the model each physical reader with the door it is associated to. Therefore, it is not necessary to have a particular set for the readers. The set of blocked readers is represented by a subset of doors which we will name BLR (for blocked readers): this is invariant **inv3_1** below.

The messages which are sent from the readers towards the control system are "door-person" pairs represented collectively by the variable $mCard$ (each one of these pairs $d \mapsto p$ represents what the reader associated with the door d has read on the card of the person p). They make up a partial function from the set D of doors to the set P of persons (invariant **inv3_2**): intuitively, this comes from the fact that no reader can implicate more than one person in the messages it sends because its slot is obstructed as seen above. However, this is not an injective function because nothing prevents a person from sliding his card into another reader when he has not gone through the door associated to the first one and the 30 seconds corresponding to this are not over (this is a case of strange behavior that cannot be eliminated).

Lastly, the set of acknowledgement messages is represented by the set $mAckn$, which is therefore a subset of doors (invariant **inv3_3**). These elements are formally defined as follows:

```
variables:  ...
            $BLR$
            $mCard$
            $mAckn$
```

inv3_1 :	$BLR \subseteq D$
inv3_2 :	$mCard \in D \rightarrowtail P$
inv3_3 :	$mAckn \subseteq D$

While a reader is obstructed, the door in question is in one of the four following *exclusive* situations : (1) it is consigned in an input message as yet untreated by the system, (2) its green light is on, (3) its red light is on, (4) it is consigned in an acknowledgement message as yet untreated by the reader. These different states characterize the progression of the information through the system. They correspond to the following supplementary invariants:

inv3_4 :	$\mathsf{dom}\,(mCard) \cup grn \cup red \cup mAckn = BLR$
inv3_5 :	$\mathsf{dom}\,(mCard) \cap (grn \cup red \cup mAckn) = \varnothing$
inv3_6 :	$mAckn \cap (grn \cup red) = \varnothing$

Since we already know that the sets grn and red are disjoint (this is invariant **inv2_7**), we can say that the four sets $\mathsf{dom}\,(mCard)$, grn, red, and $mAckn$ form a *partition* of the set BLR.

6.4 Events

The new event we are introducing at this stage is the one which corresponds to the reading of a card. It is a "physical" event:

```
CARD
  any $p, d$ where
    $p \in P$
    $d \in D \setminus BLR$
  then
    $BLR := BLR \cup \{d\}$
    $mCard := mCard \cup \{d \mapsto p\}$
  end
```

Note the the guard $d \in D \setminus BLR$ indicates that no card can be introduced in the reader while it is blocked. This is a "physical" guard.

We can now find the refinements of two events accept and refuse. They are almost identical to their previous versions except that now the implied elements p and d are those read on a message coming from a card reader :

```
accept
  any $p, d$ where
    $d \mapsto p \in mCard$
    $sit(p) = org(d)$
    $p \mapsto dst(d) \in aut$
    $p \notin \mathrm{dom}\,(dap)$
  then
    $dap(p) := d$
    $grn := grn \cup \{d\}$
    $mCard := mCard \setminus \{d \mapsto p\}$
  end
```

```
refuse
  any $p, d$ where
    $d \mapsto p \in mCard$
    $\neg\,(\, sit(p) = org(d)$
        $p \mapsto dst(d) \in aut$
        $p \notin \mathrm{dom}\,(dap)\,)$
  then
    $red := red \cup \{d\}$
    $mCard := mCard \setminus \{d \mapsto p\}$
  end
```

Note that the message $d \mapsto p$, once read, is removed from the "channel" $mCard$. The event pass is almost identical to its previous versions. The reading of the card is only confirmed by the dispatching of the corresponding acknowledgement message towards the corresponding reader by means of the "channel" $mAckn$.

```
pass
  any $d$ where
    $d \in grn$
  then
    $sit(dap^{-1}(d)) := dst(d)$
    $dap := dap \rhd\!\!\!\!- \{d\}$
    $grn := grn \setminus \{d\}$
    $mAckn := mAckn \cup \{d\}$
  end
```

Likewise the two events off_grn and off_red also contain the dispatching of an acknowledgement message to the card reader.

```
off_grn
  any $d$ where
    $d \in grn$
  then
    $dap := dap \rhd\!\!\!\!- \{d\}$
    $grn := grn \setminus \{d\}$
    $mAckn := mAckn \cup \{d\}$
  end
```

```
off_red
  any $d$ where
    $d \in red$
  then
    $red := red \setminus \{d\}$
    $mAckn := mAckn \cup \{d\}$
  end
```

Lastly, a new physical event ACKN closes the protocol by unblocking the corresponding reader:

```
ACKN
  any $d$ where
    $d \in mAckn$
  then
    $BLR := BLR \setminus \{d\}$
    $mAckn := mAckn \setminus \{d\}$
  end
```

6.5 Synchronization

The various events of this refinement are now synchronized as follows:

$$\begin{tikzcd}
 & & & \text{CARD} \arrow[dll] \arrow[dr] & \\
 & \text{accept} \arrow[dl] \arrow[dr] & & & \text{refuse} \arrow[d] \\
\text{pass} \arrow[drr] & & \text{off\_grn} \arrow[d] & & \text{off\_red} \arrow[dll] \\
 & & \text{ACKN} & &
\end{tikzcd}$$

6.6 Proofs

The proof of this refinement does not induce any particular problem.

7 Fourth Refinement

7.1 Decisions Concerning the Physical Doors

We are now introducing the physical controls of the door (blocking and acceptance) as well as those for the detection of pass. We are also taking the following decision concerning “local” behavior of doors: when a door has been cleared, it blocks itself automatically without any intervention from the control system.

We are also making another important decision, which is the following: it is supposed that each door incorporates a local clock which assures temporized blocking after 30 seconds together with the extinction of the green light, or the extinction of the red light after 2 seconds.

7.2 Variables and Invariant: the Green Chain

The formalization takes place thanks to a certain number of new variables. First the set $mAccept$ of messages sent by the control system to accept the doors. As we have seen above with the card readers, the set of accepted doors is introduced: it is called GRN since it corresponds to the *doors whose green light is physically on*. Then we have the set $mPass$ of messages sent by each door after detection of clearing. Lastly, we have the set $mOff\_grn$ of messages sent by the doors to signal automatic re-blocking after 30 seconds. The following invariant is obtained:

variables: $\ldots$
GRN
$mAccept$
$mOff\_grn$
$mPass$

These sets are exclusive and their union is equal to the set grn of doors whose *green light is logically on*. As in the previous section, these properties show the progression of the information. Also notice that GRN is included in $mAccept$.

inv4_1 : $mAccept \cup mPass \cup mOff\_grn = grn$

inv4_2 : $mAccept \cap (mPass \cup mOff\_grn) = \varnothing$

inv4_3 : $mPass \cap mOff\_grn = \varnothing$

inv4_4 : $GRN \subseteq mAccept$

As can be seen, it is possible for a door to be logically green while it is not yet or not any more physically green. Moreover, invariant **inv4_5** makes the variable grn useless in this refinement.

7.3 Variables and Invariant: the Red Chain

In a completely symmetrical way to the "green chain", we are now going to study the "red chain". The following variables are to be found. First the set $mRefuse$ of messages used to send to a door the order to put on its red light. Then the set RED of doors whose red light is physically on. Finally we have the set of messages $mOff\_red$ used for sending the corresponding information to the part of the software concerned with the extinction of the red light. These last messages are sent automatically by the door 2 seconds after the red light goes on. The following invariant is obtained :

variables: $\ldots$
RED
$mRefuse$
$mOff\_red$

These sets are exclusive and their union is equal to the set red of doors whose red light is logically on. As before, these properties show the progression of information. Moreover RED is included in $mRefuse$.

inv4_5 : $mRefuse \cup mOff\_red = red$

inv4_6 : $mRefuse \cap mOff\_red = \varnothing$

inv4_7 : $RED \subseteq mRefuse$

As can be seen, it is possible for a door to be logically red while it is not yet or not any more physically red. Moreover, invariant **inv4_12** makes the variable red useless in this refinement.

7.4 The Events

Let us now consider the events. Those which correspond to card readers which have not been changed in this refinement will not be copied here. Inversely, the event accept has been slightly modified . The sending of a physical acceptance message to the doors has been added :

```
accept
  any $p, d$ where
    $d \mapsto p \in mCard$
    $sit(p) = org(d)$
    $p \mapsto dst(d) \in aut$
    $p \notin \mathsf{dom}\,(dap)$
  then
    $dap(p) := d$
    $mCard := mCard \setminus \{d \mapsto p\}$
    $mAccept := mAccept \cup \{d\}$
  end
```

We now find the physical event of acceptance to a door and the physical lighting of a green light :

```
ACCEPT
  any $d$ where
    $d \in mAccept$
  then
    $GRN := GRN \cup \{d\}$
  end
```

It is interesting to remark the gap between the logical acceptance of the door (accept event in the software) and the physical acceptance (event ACCEPT of the hardware). This gap evokes a major problem of distributed systems which is that of distinguishing between the intention (software) and the real action (hardware). We now find the physical event corresponding to the clearing of the door. It is to be noted that the door does not “know” who is clearing it.

```
PASS
  any $d$ where
    $d \in GRN$
  then
    $GRN := GRN \setminus \{d\}$
    $mPass := mPass \cup \{d\}$
    $mAccept := mAccept \setminus \{d\}$
  end
```

This physical pass is followed by a logical pass which is almost identical to its version during the previous refinement. The only difference corresponds to the fact that the launching of this event is now due to the reception of a message. It is to be noted here that the event pass “ knows” who is passing: we are dealing with the person implied in acceptance of the door. Once again we can remark a gap between physical detection (PASS event of the hardware) and its logical effect (pass event of the software).

```
pass
  any $d$ where
    $d \in mPass$
  then
    $sit(dap^{-1}(d)) := dst(d)$
    $dap := dap \rhd \{d\}$
    $mAckn := mAckn \cup \{d\}$
    $mPass := mPass \setminus \{d\}$
  end
```

The event which consists in physically blocking the door (from a clock supposedly inside the door which starts up 30 seconds after its acceptance if no-one has cleared it in the meantime) is the following. The message of re-blocking is sent to the software.

```
OFF_GRN
  any $d$ where
    $d \in GRN$
  then
    $GRN := GRN \setminus \{d\}$
    $mOff\_grn := mOff\_grn \cup \{d\}$
    $mAccept := mAccept \setminus \{d\}$
  end
```

Finally, we can find a new version of the event off_grn, which gets underway on reception of the previous message.

```
off_grn
  any $d$ where
    $d \in mOff\_grn$
  then
    $dap := dap \rhd \{d\}$
    $mAckn := mAckn \cup \{d\}$
    $mOff\_grn := mOff\_grn \setminus \{d\}$
  end
```

The event refuse is slightly modified so as to allow a message concerning the lighting of the red light to be sent.

```
refuse
  any $p, d$ where
    $d \mapsto p \in mCard$
    $\neg\, (\, sit(p) = org(d)$
        $p \mapsto dst(d) \in aut$
        $p \notin \mathrm{dom}\,(dap)\,)$
  then
    $mCard := mCard \setminus \{q \mapsto p\}$
    $mRefuse := mRefuse \cup \{q\}$
  end
```

The first hardware event after this corresponds to the reception of the previous message and the effective lighting up of the red light. The automatic extinction of the red light after 2 seconds corresponds to the following second event which sends a message to the software so as to warn it.

```
REFUSE
  any $d$ where
    $d \in mRefuse$
  then
    $RED := RED \cup \{d\}$
  end
```

```
OFF_RED
  any $d$ where
    $d \in RED$
  then
    $RED := RED \setminus \{d\}$
    $mOff\_red := mOff\_red \cup \{d\}$
    $mRefuse := mRefuse \setminus \{d\}$
  end
```

The event off_red is slightly modified as regards its previous version: it is now set off by the reception of the previous message:

```
off_red
  any $d$ where
    $d \in mOff\_red$
  then
    $mAckn := mAckn \cup \{d\}$
    $mOff\_red := mOff\_red \setminus \{d\}$
  end
```

7.5 Synchronization

We now obtain the following complete synchronization between the software and the hardware :

$$\begin{tikzcd}
 & & \text{CARD} \arrow[dl] \arrow[dr] & \\
 & \text{accept} \arrow[d] & & \text{refuse} \arrow[d] \\
 & \text{ACCEPT} \arrow[dl] \arrow[dr] & & \text{REFUSE} \arrow[d] \\
\text{PASS} \arrow[d] & & \text{OFF\_GRN} \arrow[d] & \text{OFF\_RED} \arrow[d] \\
\text{pass} \arrow[drr] & & \text{off\_grn} \arrow[d] & \text{off\_red} \arrow[dl] \\
 & & \text{ACKN} &
\end{tikzcd}$$