UNIVERSITY OF Southampton
School of Electronics and Computer Science

The Rodin Platform for Incremental Modelling in Event-B

Stefan Hallerstede
University of Southampton

FMCO 2008, 22/10/2008

www.deploy-project.eu

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Outline

- Complex Systems Modelling in Event-B
- Interplay of proof and modelling
- Worked example: Access to secure building
- Tool Animation
- Conclusion

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Design of complex systems

- Need for rigorous modelling
- The main purpose of modelling is reasoning
  - to improve our understanding of the system
  - to clarify assumptions about the system
  - to increase quality of system model
- Need for refinement
  - Too many details to address final system directly

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Reasoning about complex systems

- Requires rigorous reasoning
  - Formal notation
  - Formal reasoning
- This is offered by formal proof
  - Why prove?
    - To incrementally improve a model
    - To take advantage of failing proofs
  - What to prove?
    - Proof obligations associated with each model

UNIVERSITY OF
Southampton
School of Electronics
and Computer Science

List of core Event-B proof obligations

- Feasibility of events
- Invariant preservation by events
- Refinement of events
- Introduction of new events
- Convergence of events
- Enabledness of events

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Outline

- Complex Systems Modelling in Event-B
- Interplay of proof and modelling
- Worked example: Access to secure building
- Tool Animation
- Conclusion

Example of an Event-B machine

invariants

$inv1 : auth \in User \leftrightarrow Room$

$inv2 : in \in User \nrightarrow Room$

$inv3 : in \subseteq auth$

event *enter*

any

u, r

when

$grd1 : u \notin \mathrm{dom}(in)$

$grd2 : u \mapsto r \in auth$

then

$act1 : in := in \cup \{u \mapsto r\}$

end

parameters

guards

actions

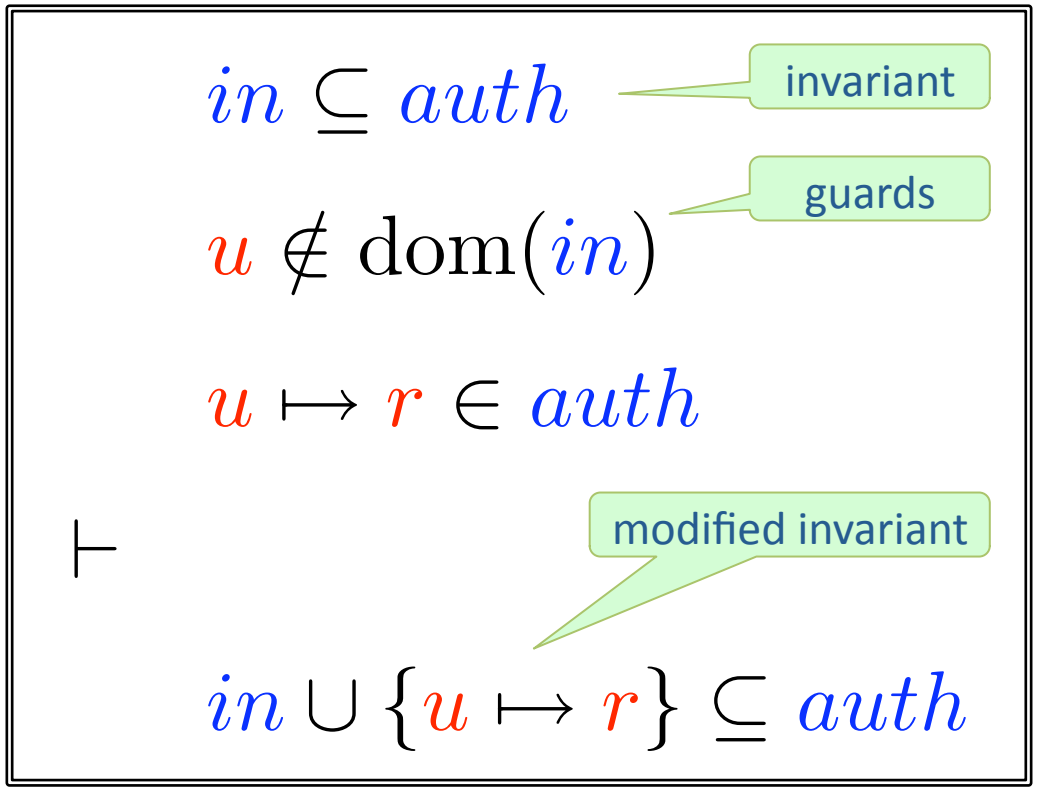
Invariant properties:

- *inv1* : A user is authorised to be in certain rooms
- *inv2* : A user can be at most in one room
- *inv3* : A user can only be in rooms where he is authorised to be

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Preservation of $in \subseteq auth$ by event *enter*

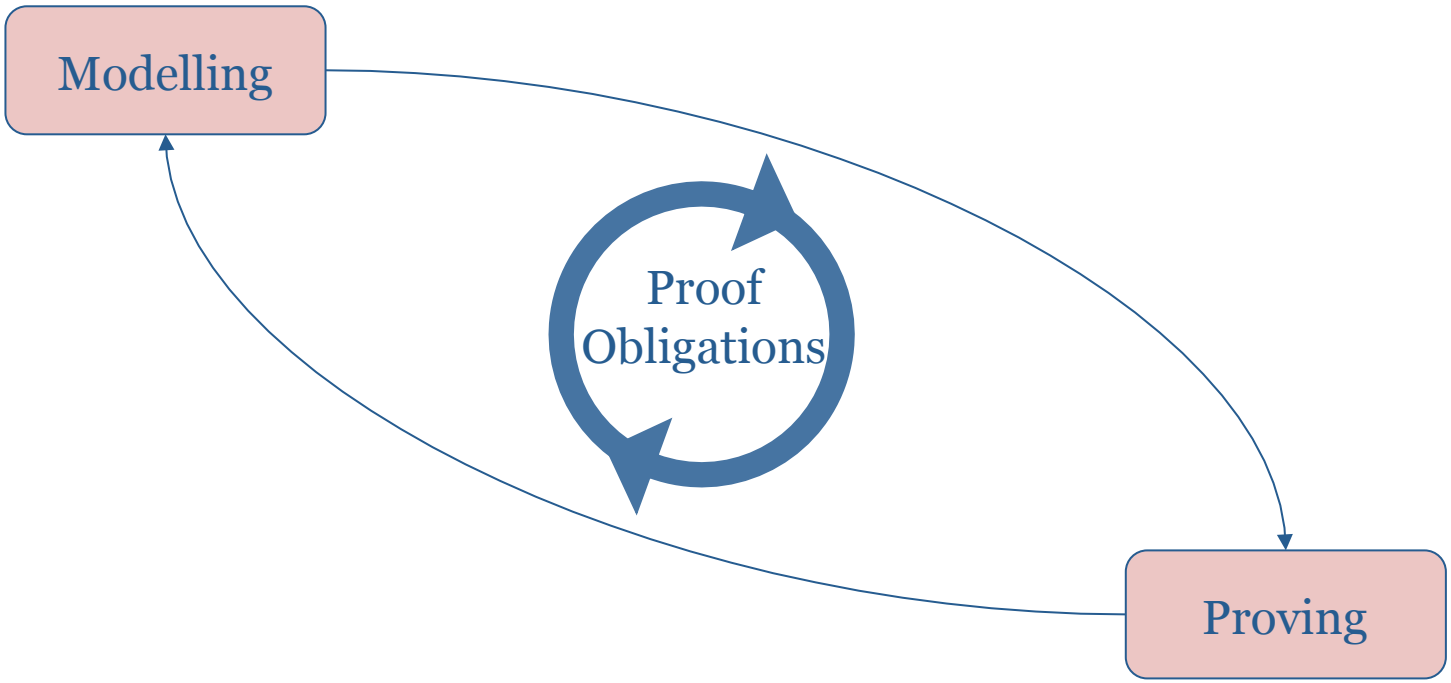
- Proof obligation:



UNIVERSITY OF Southampton
School of Electronics and Computer Science

Use of proof obligations for modelling

- Modelling is an incremental activity



- Proof obligations are automatically generated

Creating a model incrementally

- What does this mean?
- We do not demonstrate the actual tool "Rodin"
  - But focus on essential features
  - Removed everything that could distract
- Aim:
  - To illustrate method and tool
  - Not to get distracted by features of the Rodin tool
- By way of an example
  - Access to a secure building

Outline

- Complex Systems Modelling in Event-B
- Interplay of proof and modelling
- Worked example: Access to secure building
- Tool Animation
- Conclusion

UNIVERSITY OF Southampton
School of Electronics and Computer Science

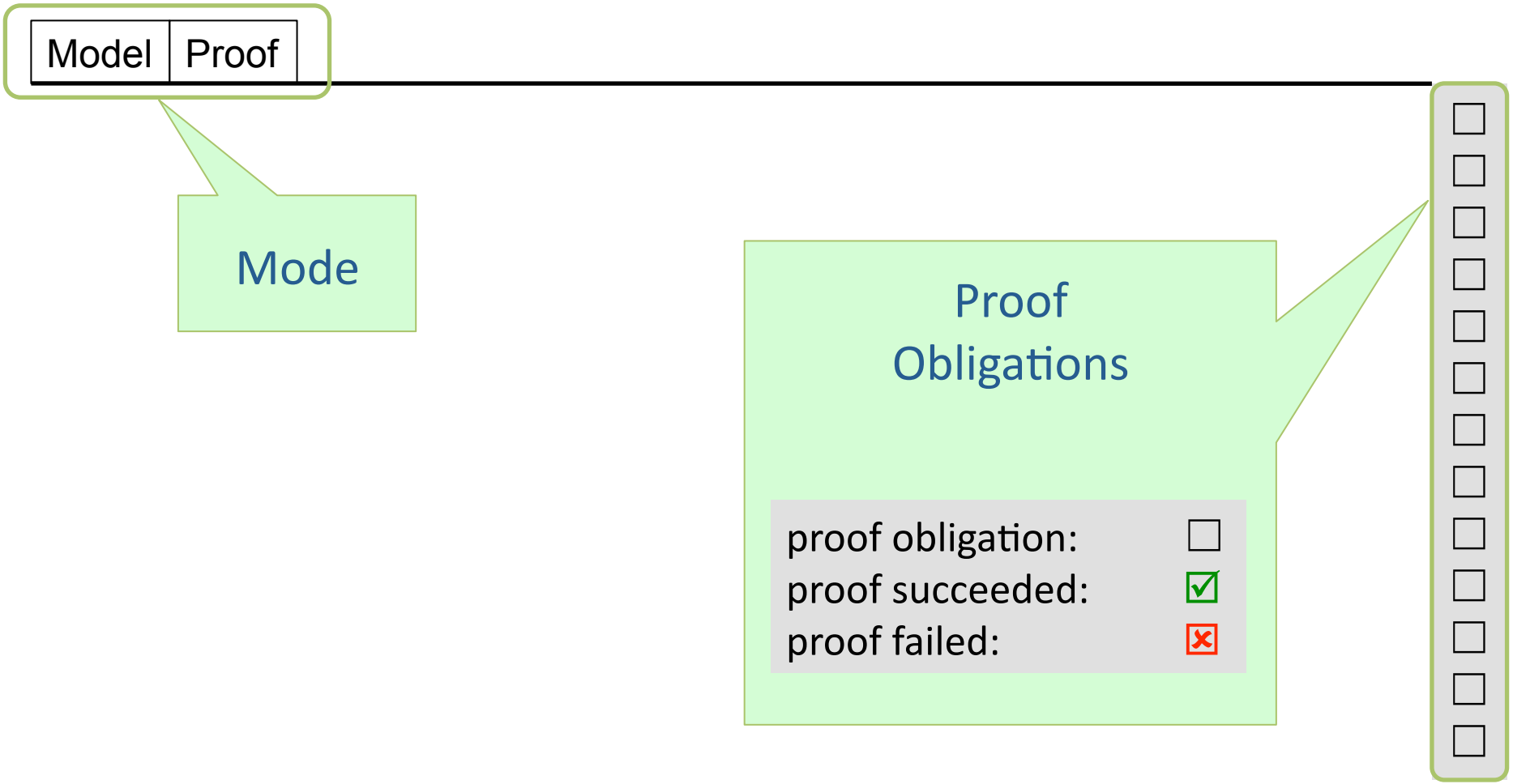
Description of the secure access model

- Abstract model
  - *Users*, *Rooms*
  - Entering/leaving rooms; adding/removing authorisations
- Refined model
  - *Tokens*
  - Data-refinement of abstract model
- The model itself is not of importance
  - But the way we create it is important

Focus on three modelling scenarios

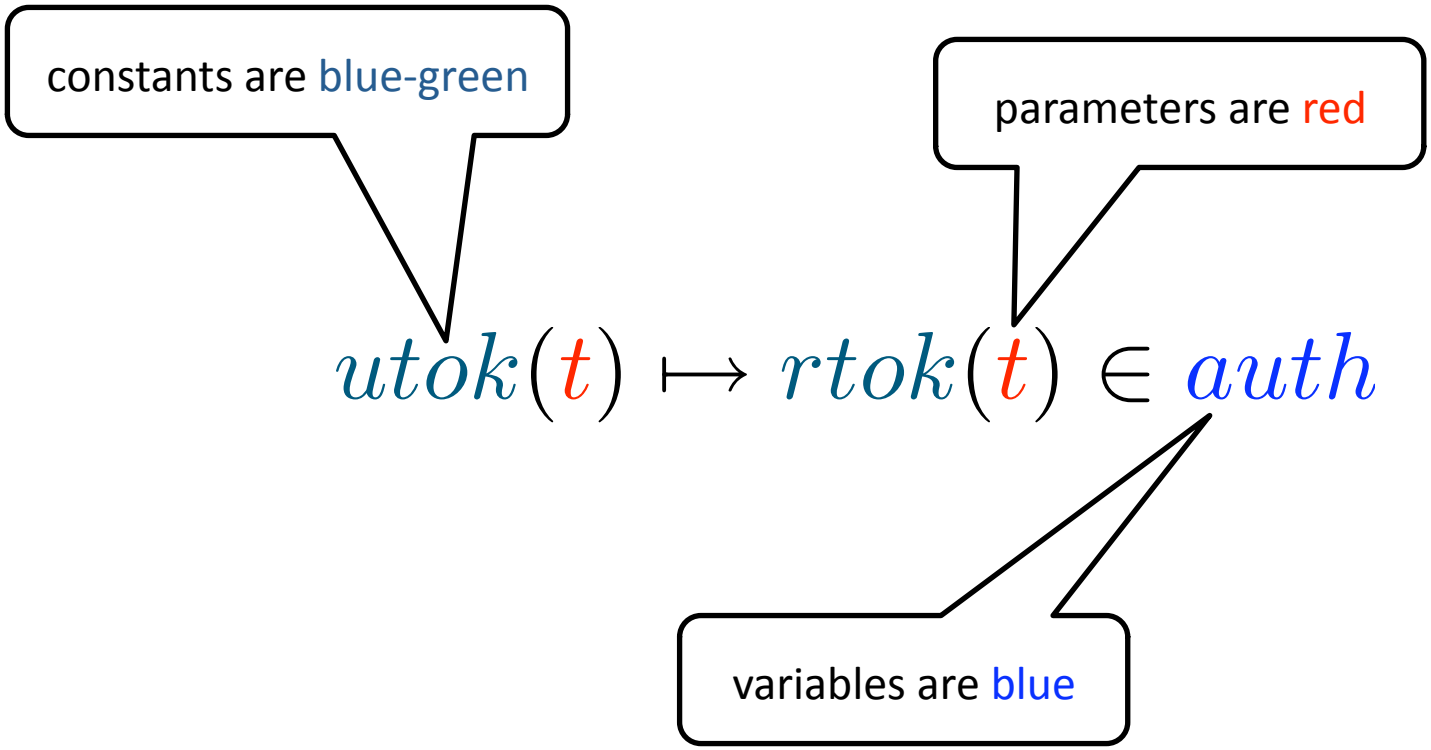
- Adding events
  1. Reasoning about guards and invariants
- Refining an event
  2. Reasoning about parameters
  3. Reasoning about guards and invariants

Layout of the modelling canvas



Colouring conventions for formulas

Model | Proof



UNIVERSITY OF Southampton
School of Electronics and Computer Science

Creation of the **abstract model**

Model | Proof

Summary:

- Variables: *in*, *auth*
- Events:
  - *enter* – enter building
  - *leave* – leave building
  - *addAuth* – add authorisation
  - *remAuth* – remove authorisation

Location and authorisation of users

Model | Proof

invariants

$inv1 : auth \in User \leftrightarrow Room$

$inv2 : in \in User \nrightarrow Room$

$inv3 : in \subseteq auth$

Location and authorisation of users

Model | Proof

invariants

$inv1 : auth \in User \leftrightarrow Room$

$inv2 : in \in User \nrightarrow Room$

$inv3 : in \subseteq auth$

initialisation

$act1 : in := \emptyset$

$act2 : auth := \emptyset$

Location and authorisation of users

Model | Proof

invariants

$inv1 : auth \in User \leftrightarrow Room$

$inv2 : in \in User \nrightarrow Room$

$inv3 : in \subseteq auth$

initialisation

$act1 : in := \emptyset$

$act2 : auth := \emptyset$

Location and authorisation of users

Model | Proof

invariants

$inv1 : auth \in User \leftrightarrow Room$

$inv2 : in \in User \nrightarrow Room$

$inv3 : in \subseteq auth$

initialisation

$act1 : in := \emptyset$

$act2 : auth := \emptyset$

☑
☑
☑

Entering a room

Model | Proof

event $enter$
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in auth$
  then
    $act1 : in := in \cup \{u \mapsto r\}$
  end

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Entering a room

Model | Proof

event *enter*
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in auth$
  then
    $act1 : in := in \cup \{u \mapsto r\}$
  end

☑
☑
☑
☐
☐

Proof obligation:

Preservation of invariant *inv3*

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Entering a room

Model | Proof

Preservation of invariant *inv3*

$in \subseteq auth$

$u \notin \mathrm{dom}(in)$

$u \mapsto r \in auth$

$\vdash$

$in \cup \{u \mapsto r\} \subseteq auth$

☑
☑
☑
☐
☐

Proof obligation:

Preservation of invariant *inv3*

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Entering a room

Model | Proof

Preservation of invariant *inv3*

$in \subseteq auth$

$u \notin \mathrm{dom}(in)$

$u \mapsto r \in auth$

$\vdash$

$in \cup \{u \mapsto r\} \subseteq auth$

☑
☑
☑
☐
☑

Proof obligation:

Preservation of invariant *inv3*

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Entering a room

Model | Proof

event $enter$
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in auth$
  then
    $act1 : in := in \cup \{u \mapsto r\}$
  end

☑
☑
☑
☐
☑

Entering a room

Model | Proof

event $enter$
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in auth$
  then
    $act1 : in := in \cup \{u \mapsto r\}$
  end

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Leaving a room

Model | Proof

event $leave$
  any
    u
  when
    $grd1 : u \in \mathrm{dom}(in)$
  then
    $act1 : in := \{u\} \lhd in$
  end

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Leaving a room

Model | Proof

event $leave$
  any
    u
  when
    $grd1 : u \in \mathrm{dom}(in)$
  then
    $act1 : in := \{u\} \lhd in$
  end

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Leaving a room

Model | Proof

event $leave$
  any
    u
  when
    $grd1 : u \in \mathrm{dom}(in)$
  then
    $act1 : in := \{u\} \lhd in$
  end

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Adding an authorisation

Model | Proof

event $addAuth$
  any
    u, r
  when
    $grd1 : u \in User$
    $grd2 : r \in Room$
  then
    $act1 : auth := auth \cup \{u \mapsto r\}$
  end

Adding an authorisation

Model | Proof

event $addAuth$
  any
    u, r
  when
    $grd1 : u \in User$
    $grd2 : r \in Room$
  then
    $act1 : auth := auth \cup \{u \mapsto r\}$
  end

☑
☑
☑
☑
☑
☑
☑
☐
☐

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Adding an authorisation

Model | Proof

event $addAuth$
  any
    u, r
  when
    $grd1 : u \in User$
    $grd2 : r \in Room$
  then
    $act1 : auth := auth \cup \{u \mapsto r\}$
  end

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Removing an authorisation

Model | Proof

event $remAuth$
  any
    u, r
  when
    $grd1 : u \in User$
    $grd2 : r \in Room$
  then
    $act1 : auth := auth \setminus \{u \mapsto r\}$
  end

Removing an authorisation

Model | Proof

event $remAuth$
  any
    u, r
  when
    $grd1 : u \in User$
    $grd2 : r \in Room$
  then
    $act1 : auth := auth \setminus \{u \mapsto r\}$
  end

Removing an authorisation

Model | Proof

event $remAuth$
  any
    u, r
  when
    $grd1 : u \in User$
    $grd2 : r \in Room$
  then
    $act1 : auth := auth \setminus \{u \mapsto r\}$
  end

Proof obligation:

Preservation of invariant *inv3*

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Removing an authorisation

Model | Proof

Preservation of invariant *inv3*

$in \subseteq auth$

$u \in User$

$r \in Room$

$\vdash$

$in \subseteq auth \setminus \{u \mapsto r\}$

Possible remedies:

- Modify action:
  → Remove user *u* from building
- Modify guard:
  → Require user *u* not in building
- Modify guard:
  → Require user *u* not in room *r*

$u \mapsto r \notin in$

☑ ☑ ☑ ☑ ☑ ☑ ☑ ☑ ☑ ☑ ☒

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Removing an authorisation

Model | Proof

event $remAuth$
  any
    u, r
  when
    $grd1 : u \in User$
    $grd2 : r \in Room$
  then
    $act1 : auth := auth \setminus \{u \mapsto r\}$
  end

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Removing an authorisation

Model | Proof

event $remAuth$
  any
    u, r
  when
    $grd1 : u \in User$
    $grd2 : u \mapsto r \notin in$
  then
    $act1 : auth := auth \setminus \{u \mapsto r\}$
  end

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Removing an authorisation

Model | Proof

Preservation of invariant *inv3*

$in \subseteq auth$

$u \in User$

$u \mapsto r \notin in$

$\vdash$

$in \subseteq auth \setminus \{u \mapsto r\}$

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Removing an authorisation

Model | Proof

Preservation of invariant *inv3*

$in \subseteq auth$

$u \in User$

$u \mapsto r \notin in$

$\vdash$

$in \subseteq auth \setminus \{u \mapsto r\}$

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Creation of the **refined model**

Model | Proof

Summary:

- Variables: *tok*, *auth*
- Events:
  - *enter* – enter building
  - other events not shown

Replacing Users and Rooms by Tokens

Model | Proof

axioms

$axm1 : utok \in Token \rightarrow User$

$axm2 : rtok \in Token \rightarrow Room$

Abstract model of a record type with two fields *utok* and *rtok*

Replacing Users and Rooms by Tokens

Model | Proof

invariants

$$inv4 \ : \ \forall\, u, r \cdot \\ u \mapsto r \in in \\ \Leftrightarrow \\ \exists\, t \cdot t \in tok \ \wedge \ u = utok(t) \ \wedge \ r = rtok(t)$$

Using some set-theoretic notation of Event-B *inv4* can be stated more concisely:

$$in = utok^{-1} \, ; (tok \lhd rtok)$$

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Replacing Users and Rooms by Tokens

Model | Proof

invariants

$inv4 : in = utok^{-1} ; (tok \lhd rtok)$

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Replacing Users and Rooms by Tokens

Model | Proof

invariants

$inv4 : in = utok^{-1} ; (tok \lhd rtok)$

initialisation

$act1 : tok := \emptyset$

$act2 : auth := \emptyset$

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Replacing Users and Rooms by Tokens

Model | Proof

invariants

$inv4 : in = utok^{-1} ; (tok \lhd rtok)$

initialisation

$act1 : tok := \emptyset$

$act2 : auth := \emptyset$

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Replacing Users and Rooms by Tokens

Model | Proof

invariants

$inv4 : in = utok^{-1} ; (tok \lhd rtok)$

initialisation

$act1 : tok := \emptyset$

$act2 : auth := \emptyset$

Entering a room (refined model)

Model | Proof

event *enter*
  any
    t
  when
    $grd1 : t \notin tok$
    $grd2 : utok(t) \mapsto rtok(t) \in auth$
  then
    $act1 : tok := tok \cup \{t\}$
  end

abstract event *enter*
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in auth$
  then
    $act1 : in := in \cup \{u \mapsto r\}$
  end

Entering a room (refined model)

Model | Proof

event *enter*
any
t
when
$grd1 : t \notin tok$
$grd2 : utok(t) \mapsto rtok(t) \in auth$
then
$act1 : tok := tok \cup \{t\}$
end

abstract event *enter*
any
u, r
when
$grd1 : u \notin \mathrm{dom}(in)$
$grd2 : u \mapsto r \in auth$
then
$act1 : in := in \cup \{u \mapsto r\}$
end

☑
☐
☐
☐

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Entering a room (refined model)

Model | Proof

event *enter*
  any
    t
  when
    $grd1 : t \notin tok$
    $grd2 : utok(t) \mapsto rtok(t) \in auth$
  then
    $act1 : tok := tok \cup \{t\}$
  end

abstract event *enter*
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$

$r\}$

Proof obligation:

Strengthening of guard *grd2*

Entering a room (refined model)

Model | Proof

Strengthening of guard *grd2*

$t \notin tok$

$utok(t) \mapsto rtok(t) \in auth$

$\vdash$

$u \mapsto r \in auth$

Must relate *t* to *u* and *r*

Choose witnesses for *u* and *r*:

$u = utok(t)$

$r = rtok(t)$

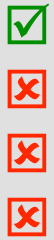
☑ ☒ ☒ ☒

Entering a room (refined model)

Model | Proof

event *enter*
  any
    t
  when
    $grd1 : t \notin tok$
    $grd2 : utok(t) \mapsto rtok(t) \in auth$
  then
    $act1 : tok := tok \cup \{t\}$
  end

abstract event *enter*
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in auth$
  then
    $act1 : in := in \cup \{u \mapsto r\}$
  end



Entering a room (refined model)

Model | Proof

event *enter*
  any
    t
  when
    $grd1 : t \notin tok$
    $grd2 : utok(t) \mapsto rtok(t) \in auth$
  with
    $u \mid u = utok(t)$
    $r \mid r = rtok(t)$
  then
    $act1 : tok := tok \cup \{t\}$
  end

abstract event *enter*
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in auth$
  then
    $act1 : in := in \cup \{u \mapsto r\}$
  end

Entering a room (refined model)

Model | Proof

Strengthening of guard *grd2*

$t \notin tok$

$utok(t) \mapsto rtok(t) \in auth$

$\vdash$

$utok(t) \mapsto rtok(t) \in auth$

Entering a room (refined model)

Model	Proof

Strengthening of guard *grd2*

$$t \notin tok$$

$$utok(t) \mapsto rtok(t) \in auth$$

$\vdash$

$$utok(t) \mapsto rtok(t) \in auth$$

☑
☑
☒
☑

Proof obligation:

Strengthening of guard *grd1*

Entering a room (refined model)

Model | Proof

event *enter*
  any
    t
  when
    $grd1 : t \notin tok$
    $grd2 : utok(t) \mapsto rtok(t) \in auth$
  with
    $u \mid u = utok(t)$
    $r \mid r = rtok(t)$
  then
    $act1 : tok := tok \cup \{t\}$
  end

abstract event *enter*
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in aut$
  then
    c ... }
  end

Proof obligation:

Strengthening of guard *grd1*

Entering a room (refined model)

Model | Proof

Strengthening of guard *grd1*

$in = utok^{-1} ; (tok \lhd rtok)$

$t \notin tok$

$utok(t) \mapsto rtok(t) \in auth$

$\vdash$

$utok(t) \notin \mathrm{dom}(in)$

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Entering a room (refined model)

Model | Proof

Strengthening of guard *grd1*

$in = utok^{-1} ; (tok \lhd rtok)$

$t \notin tok$

$utok(t) \mapsto rtok(t) \in auth$

$\vdash$

$utok(t) \notin \mathrm{dom}(utok^{-1} ; (tok \lhd rtok))$

☑
☑
☒
☑

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Entering a room (refined model)

Model | Proof

Strengthening of guard *grd1*

$$in = utok^{-1} ; (tok \lhd rtok)$$

$$t \notin tok$$

$$utok(t) \mapsto rtok(t) \in auth$$

$\vdash$

$$utok(t) \notin \mathrm{dom}((utok^{-1} \rhd tok) ; rtok)$$

rtok is total

☑ ☑ ☒ ☑

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Entering a room (refined model)

Model | Proof

Strengthening of guard *grd1*

$in = utok^{-1} ; (tok \lhd rtok)$

$t \notin tok$

$utok(t) \mapsto rtok(t) \in auth$

$\vdash$

$utok(t) \notin \mathrm{dom}(utok^{-1} \rhd tok)$

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Entering a room (refined model)

Model | Proof

Strengthening of guard *grd1*

$in = utok^{-1} ; (tok \lhd rtok)$

$t \notin tok$

$utok(t) \mapsto rtok(t) \in auth$

$\vdash$

$utok(t) \notin utok[tok]$

Guard

$t \notin tok$

is too weak.

$utok(t) \notin utok[tok]$

is a better choice for guard *grd1*

Entering a room (refined model)

Model | Proof

event *enter*
  any
    t
  when
    $grd1 : t \notin tok$
    $grd2 : utok(t) \mapsto rtok(t) \in auth$
  with
    $u \mid u = utok(t)$
    $r \mid r = rtok(t)$
  then
    $act1 : tok := tok \cup \{t\}$
  end

abstract event *enter*
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in auth$
  then
    $act1 : in := in \cup \{u \mapsto r\}$
  end

☑ ☑ ☒ ☑

Entering a room (refined model)

Model | Proof

event *enter*
  any
    t
  when
    $grd1 : utok(t) \notin utok[tok]$
    $grd2 : utok(t) \mapsto rtok(t) \in auth$
  with
    $u \mid u = utok(t)$
    $r \mid r = rtok(t)$
  then
    $act1 : tok := tok \cup \{t\}$
  end

abstract event *enter*
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in auth$
  then
    $act1 : in := in \cup \{u \mapsto r\}$
  end

☑ ☑ ☒ ☑

Entering a room (refined model)

Model | Proof

event *enter*
  any
    t
  when
    $grd1 : utok(t) \notin utok[tok]$
    $grd2 : utok(t) \mapsto rtok(t) \in auth$
  with
    $u \mid u = utok(t)$
    $r \mid r = rtok(t)$
  then
    $act1 : tok := tok \cup \{t\}$
  end

abstract event *enter*
  any
    u, r
  when
    $grd1 : u \notin \mathrm{dom}(in)$
    $grd2 : u \mapsto r \in auth$
  then
    $act1 : in := in \cup \{u \mapsto r\}$
  end

☑ ☑ ☑ ☑

Outline

- Complex Systems Modelling in Event-B
- Interplay of proof and modelling
- Worked example: Access to secure building
- Tool Animation
- Conclusion

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Conclusion

- Close relationship between modelling and proving
- Proof can give hints for improvements of model
- Incremental modelling based on heuristics
  - As opposed to refinement based on proof
- There was no time to show
  - finding invariants by inspection of proofs
  - modifying an abstract model to improve a refinement

Community

- Web:
  - www.event-b.org
  - wiki.event-b.org

- Credits

  Jean-Raymond Abrial, Jens Bendisposto, Michael Butler, Dominique Cansell, Mathieu Clabaut, Kriangsak Damchoom, Fabian Fritz, Thai Son Hoang, Sonja Holl, Cliff Jones, Thierry Lecomte, Michael Leuschel, Farhad Mehta, Christophe Métayer, Renato Silva, Colin Snook, François Terrier, Laurent Voisin, . . .