How to Explain Mistakes

Stefan Hallerstede and Michael Leuschel

Universität Düsseldorf

Teaching Formal Methods 2009
6 November 2009

HEINRICH HEINE
UNIVERSITÄT
DÜSSELDORF

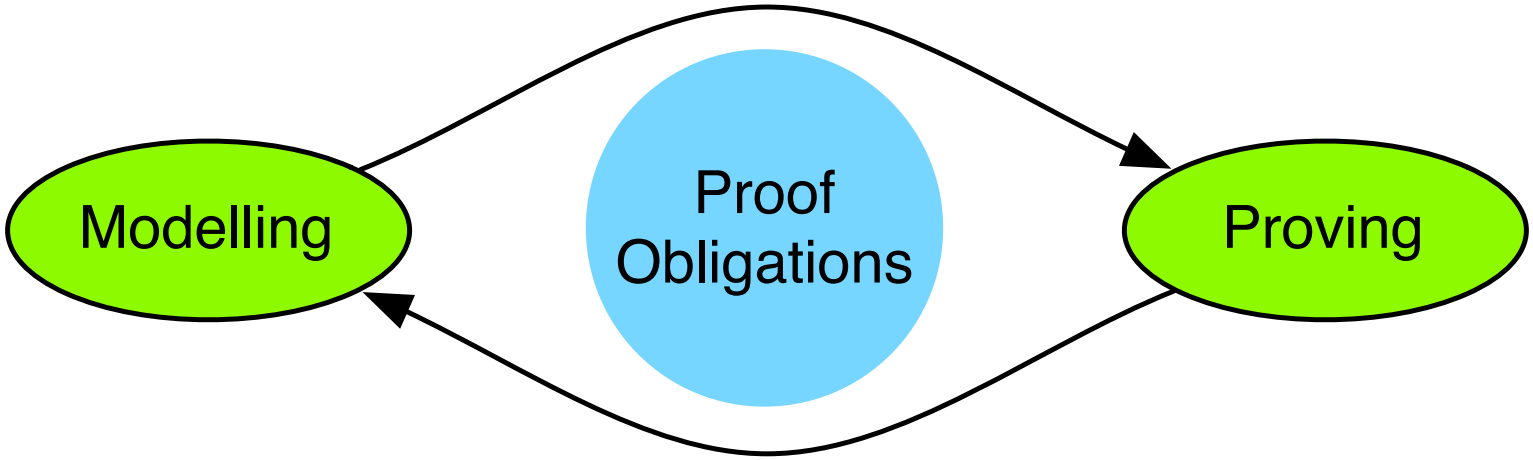
Contents

Contents

Contents

Modelling and Proving in Event-B

- Main purpose of modelling is reasoning
- Models determine what is to be formally proved
- Proof obligations are automatically generated
- Tool support is essential
- Refinement is a proof technique
- Models and proof obligations correspond closely



Contents

A Simple Example of an Event-B Model

- Invariants

  $inv1$: $auth \in Person \leftrightarrow Room$
  A person is authorised to be in certain rooms

  $inv2$: $in \in Person \nrightarrow Room$
  A person can be at most in one room

  $inv3$: $in \subseteq auth$
  A person can only be in rooms where he is authorised to be

- Events

  enter
  any
  p r
  when
  $grd1$: $p \notin \mathrm{dom}(in)$ Person is not in building
  $grd2$: $p \mapsto r \in auth$ Person is authorised to enter room
  then
  $act1$: $in := in \cup \{p \mapsto r\}$
  end

Proof Obligations of the Event-B Model

- Preservation of invariant $inv3$ by event $enter$
  - Name of proof obligation

    $$\text{“}enter/inv3/\text{INV”}$$

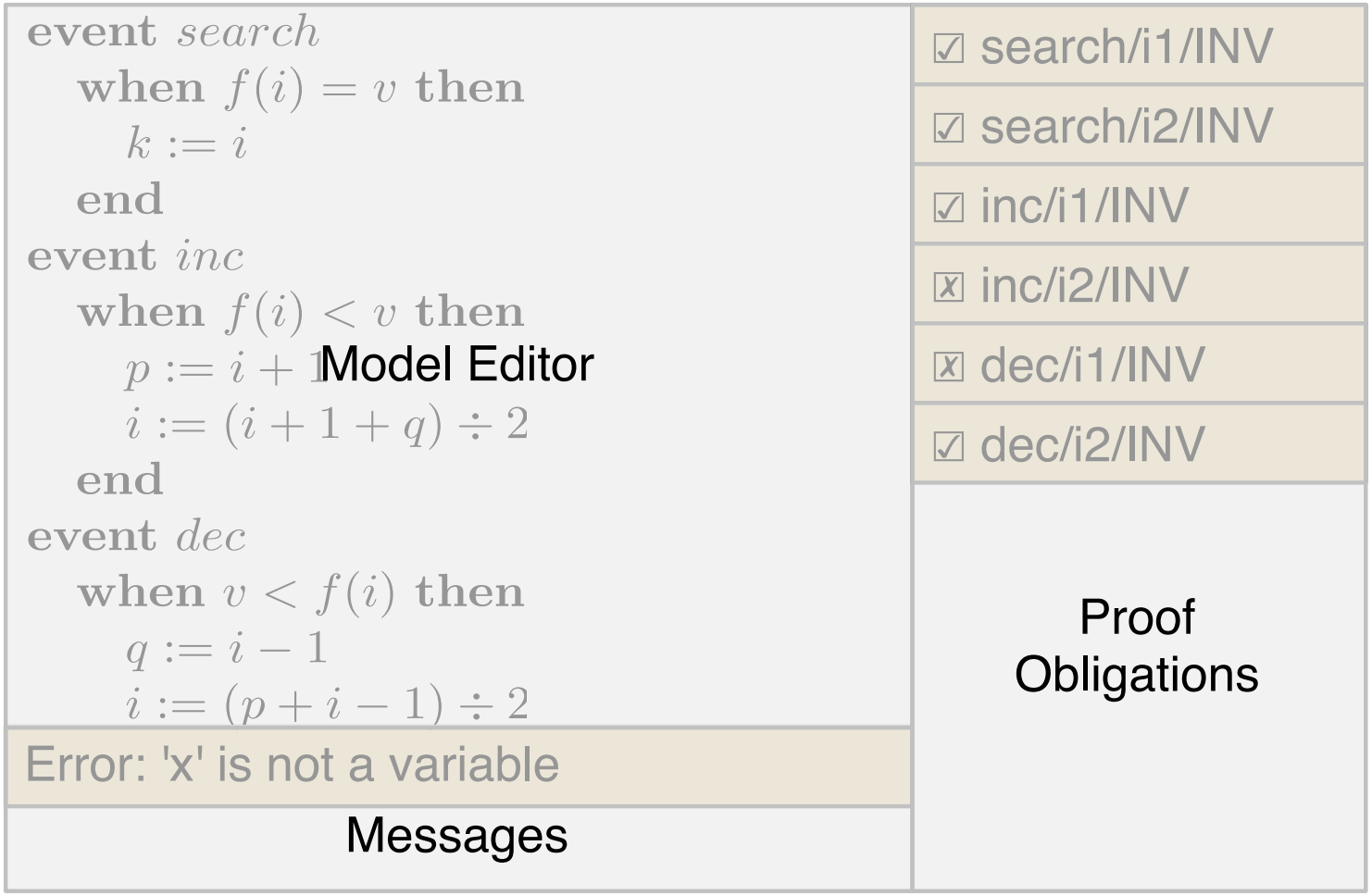
  - Sequent

$auth \in Person \leftrightarrow Room$	invariant $inv1$
$in \in Person \nrightarrow Room$	invariant $inv2$
$in \subseteq auth$	invariant $inv3$
$p \notin \mathrm{dom}(in)$	guard $grd1$
$p \mapsto r \in auth$	guard $grd2$
$\vdash$	
$in \cup \{ p \mapsto r \} \subseteq auth$	modified $(act1)$ invariant $inv3$

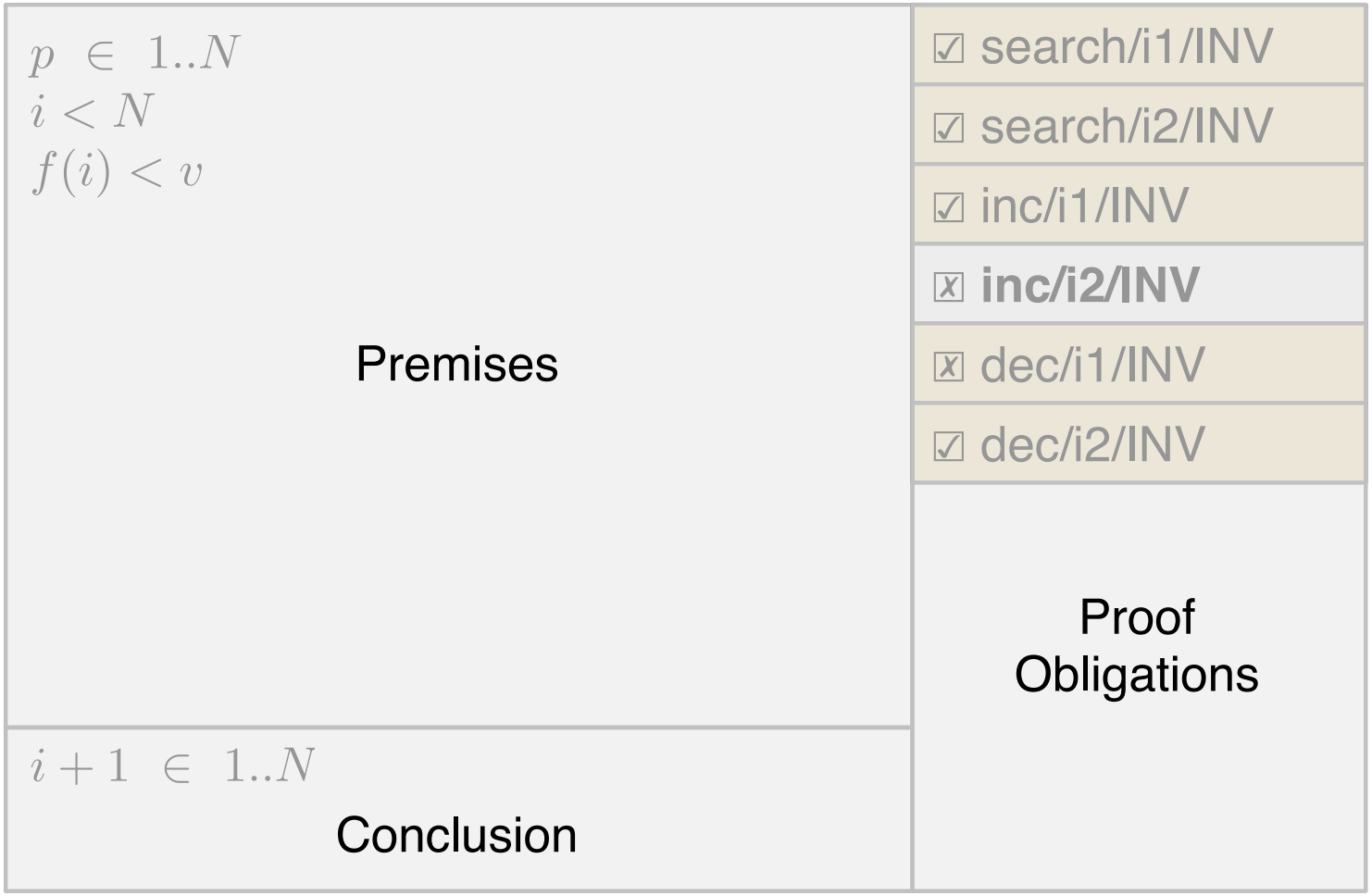
- Simple correspondence between proof obligations and model

Contents

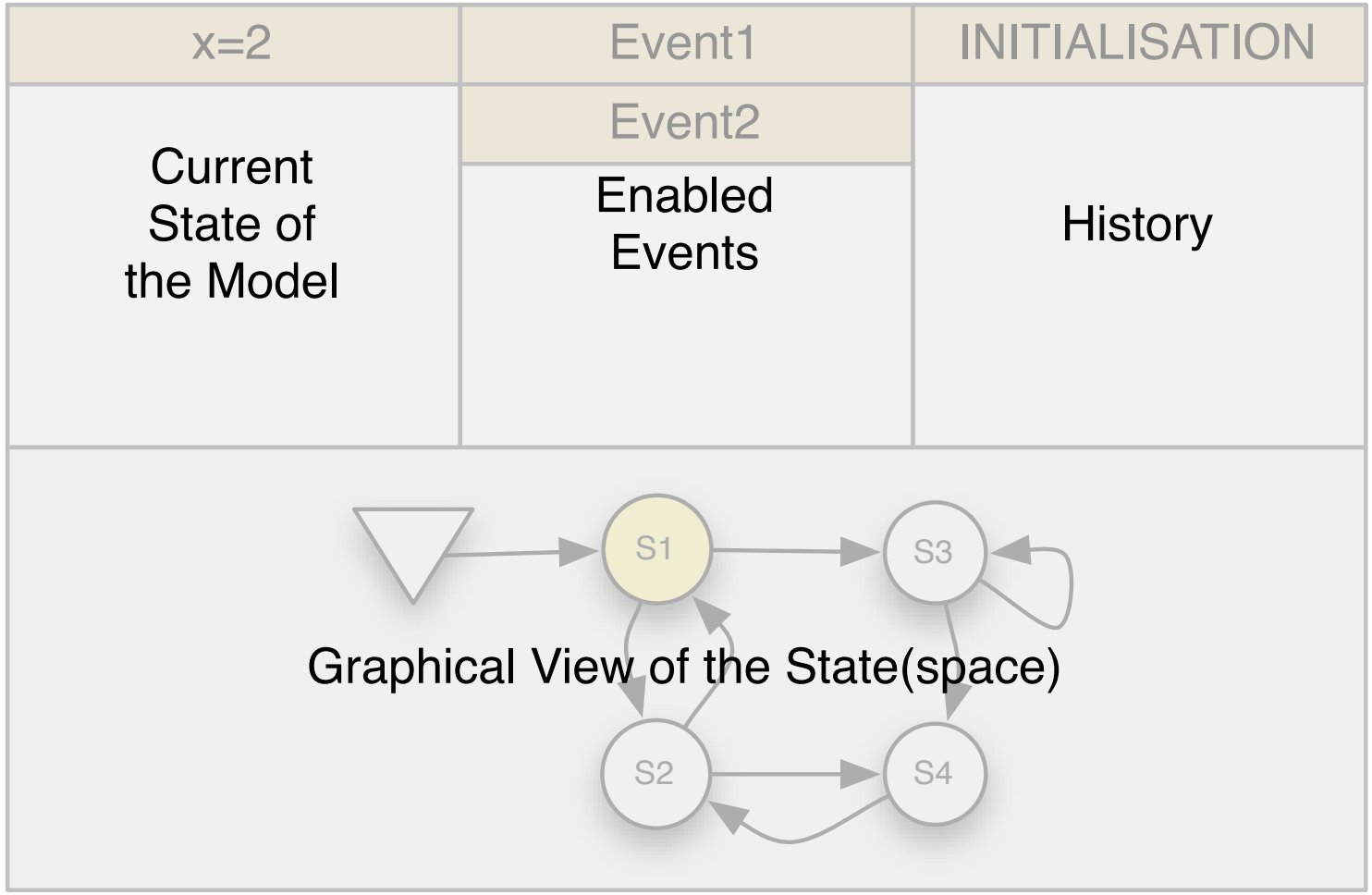
The Rodin Tool — **Modelling**



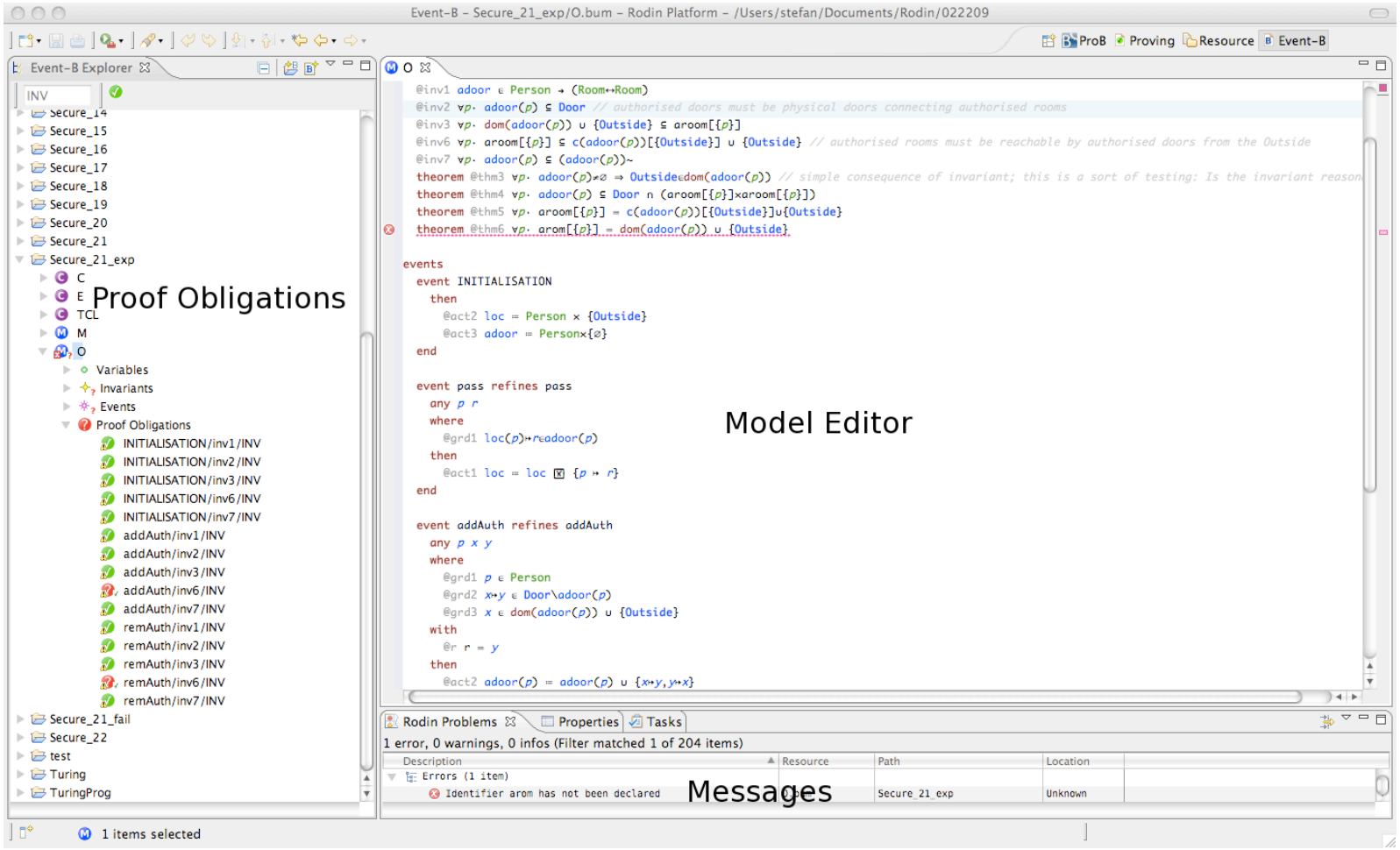
The Rodin Tool — **Proving**



The Rodin Tool — **Animation** (ProB)



Screen Shot of the Rodin Tool — **Modelling**



HEINRICH HEINE UNIVERSITÄT DÜSSELDORF

Contents

Starting From a Perfect Solutions

- Usually we present (perfect) solutions to selected problems
- This does not show how the solution was obtained
- It creates the illusion there would be a perfect solution
- This fails to demonstrate a major strength of formal methods
  - Support towards finding a good solution
  - It is not just about correctness

Finding a Good Solution

Problem solving (Pólya, Lakatos)

- Think about how to approach the problem
- Start with a model that appears reasonable
- Make mistakes
- Analyse the model
- Think again
- Improve the model
- Make mistakes

Contents

Analysing and Explaining Mistakes

- Proof is a good tool for analysing inconsistent models
- It points to the place where the inconsistency occurs
- It does not serve well for explaining inconsistencies
- Useful tools for explanation:
  - Model checking: counter examples
  - Animation: see "how it happens"
- ProB can also be used for this

Example of requirements

P1 : The system consists of persons and one building.

P2 : The building consists of rooms and doors.

P3 : Each person can be at most in one room.

P4 : Each person is authorised to be in certain rooms (but not others).

P5 : Each person is authorised to use certain doors (but not others).

P6 : Each person can only be in a room where the person is authorised to be.

P7 : Each person must be able to leave the building from any room where the person is authorised to be.

P8 : Each person can pass from one room to another if there is a door connecting the two rooms and the person has the proper authorisation.

P9 : Authorisations can be granted and revoked.

- Example provides room for misunderstanding
- Unlike a sequential program, for instance
- Model is much simplified
  from "Event Driven System Construction" by Abrial

Getting Started

- the abstract machine models room authorisations
- the concrete machine models room and door authorisations

Abstract invariants

$$inv1 : arm \in Person \leftrightarrow Room \quad \text{Property P4}$$
$$inv2 : Person \times \{\mathbf{O}\} \subseteq arm$$
$$inv3 : loc \in Person \rightarrow Room \quad \text{Property P3}$$
$$inv4 : loc \subseteq arm \quad \text{Property P6}$$

Revoking an Authorisation

P9 Authorisations can be granted and revoked.

$revoke$

any p r when

$grd1$: $p \in Person$

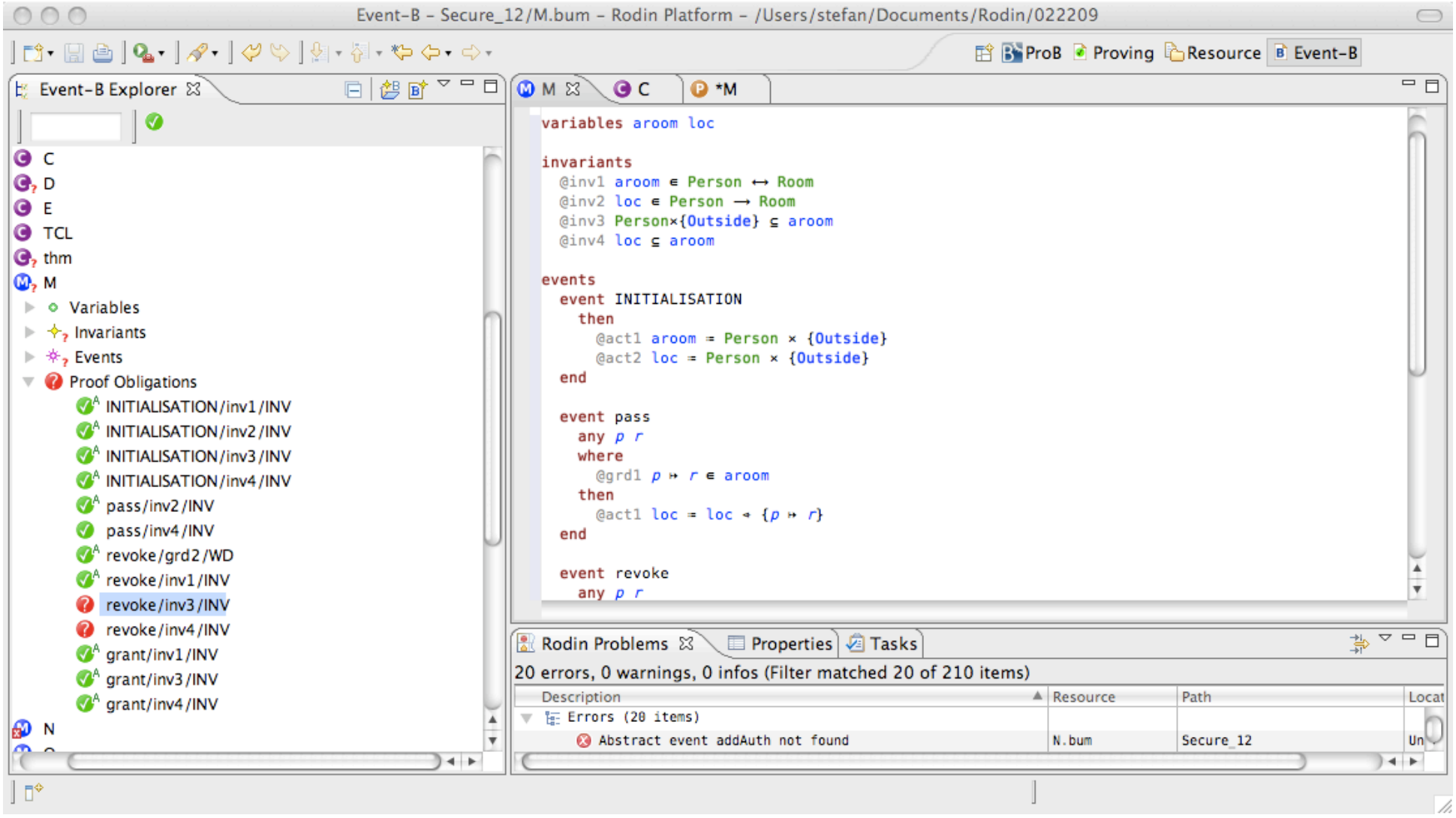
$grd2$: $p \mapsto r \notin loc$

then

$act1$: $arm := arm \setminus \{p \mapsto r\}$

end

How the model looks in Rodin



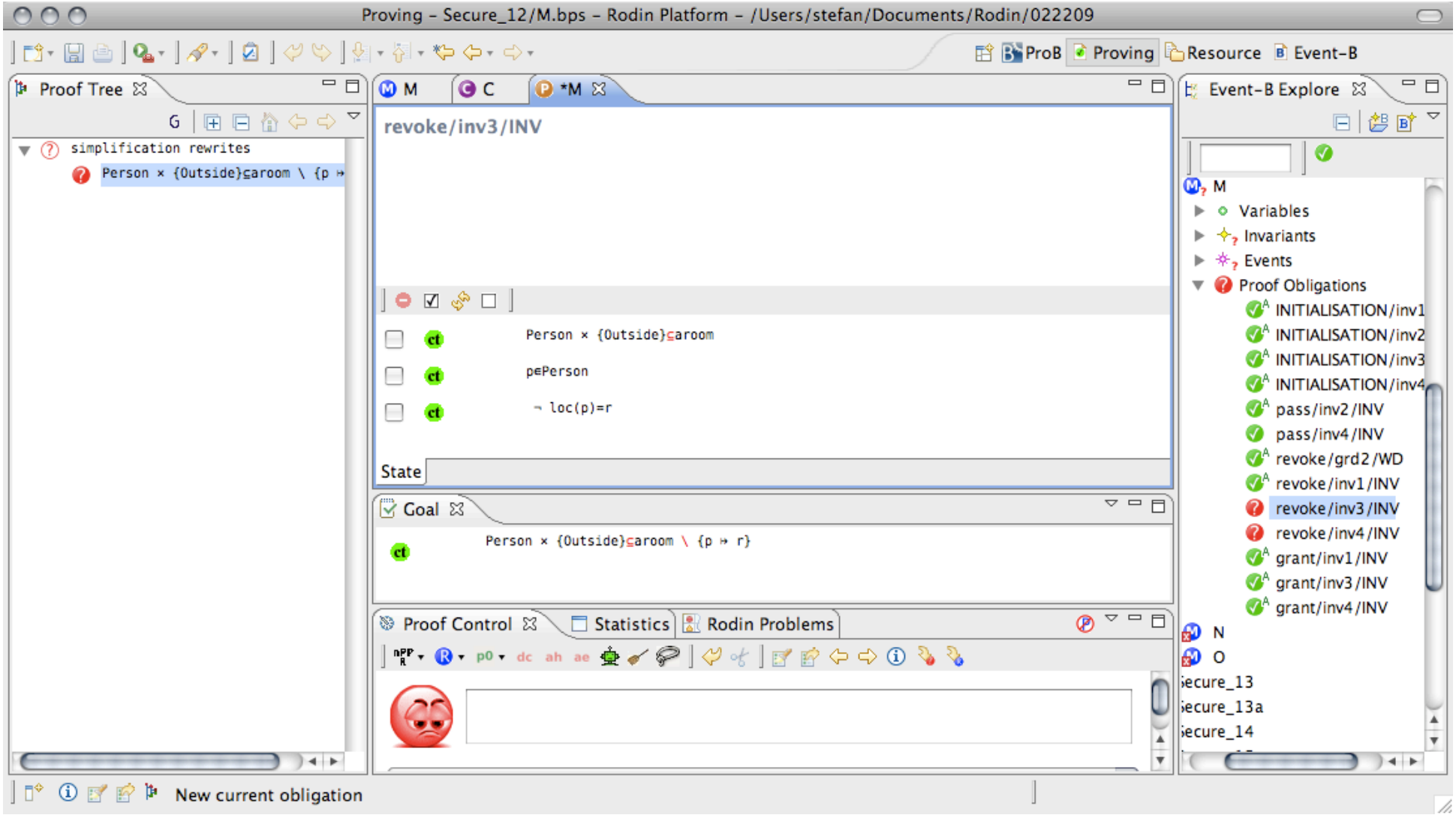
HEINRICH HEINE UNIVERSITÄT DÜSSELDORF

What have to prove:

Event $revoke$ preserves invariant $inv2$:

$$
\begin{array}{lll}
Person \times \{\mathbf{O}\} \subseteq arm & & \textit{Invariant } inv2 \\
p \in Person & & \textit{Guard } grd1 \\
p \mapsto r \notin loc & & \textit{Guard } grd2 \\
\vdash & & \\
Person \times \{\mathbf{O}\} \subseteq arm \setminus \{p \mapsto r\} & & \textit{Modified invariant } inv2
\end{array}
$$

How the proof obligation looks in Rodin



HEINRICH HEINE UNIVERSITÄT DÜSSELDORF

What have to prove:

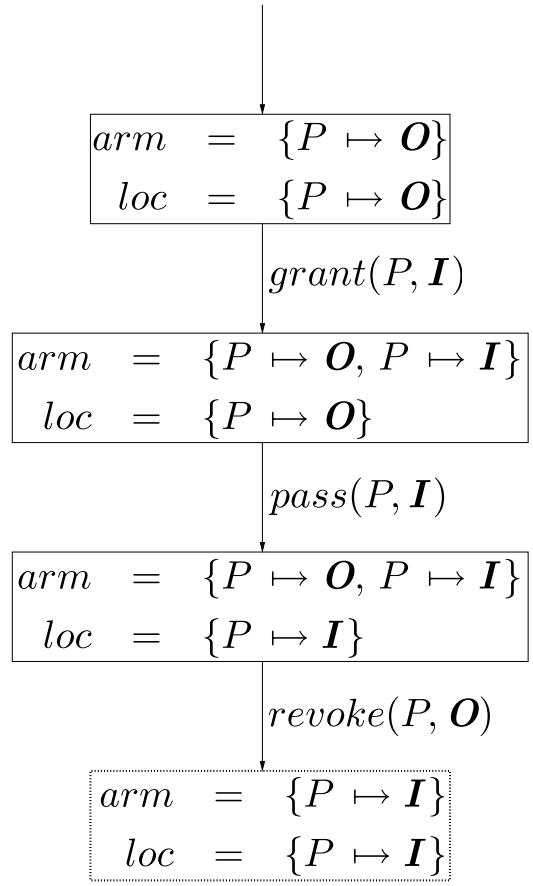
Event $revoke$ preserves invariant $inv2$:

$$
\begin{array}{ll}
Person \times \{\mathbf{O}\} \subseteq arm & \textit{Invariant } inv2 \\
p \in Person & \textit{Guard } grd1 \\
p \mapsto r \notin loc & \textit{Guard } grd2 \\
\vdash & \\
Person \times \{\mathbf{O}\} \subseteq arm \setminus \{p \mapsto r\} & \textit{Modified invariant } inv2
\end{array}
$$

- Is it (*not*) provable?
- Why?
- Our aim is to improve the model
- Not to make the proof obligation "pass"

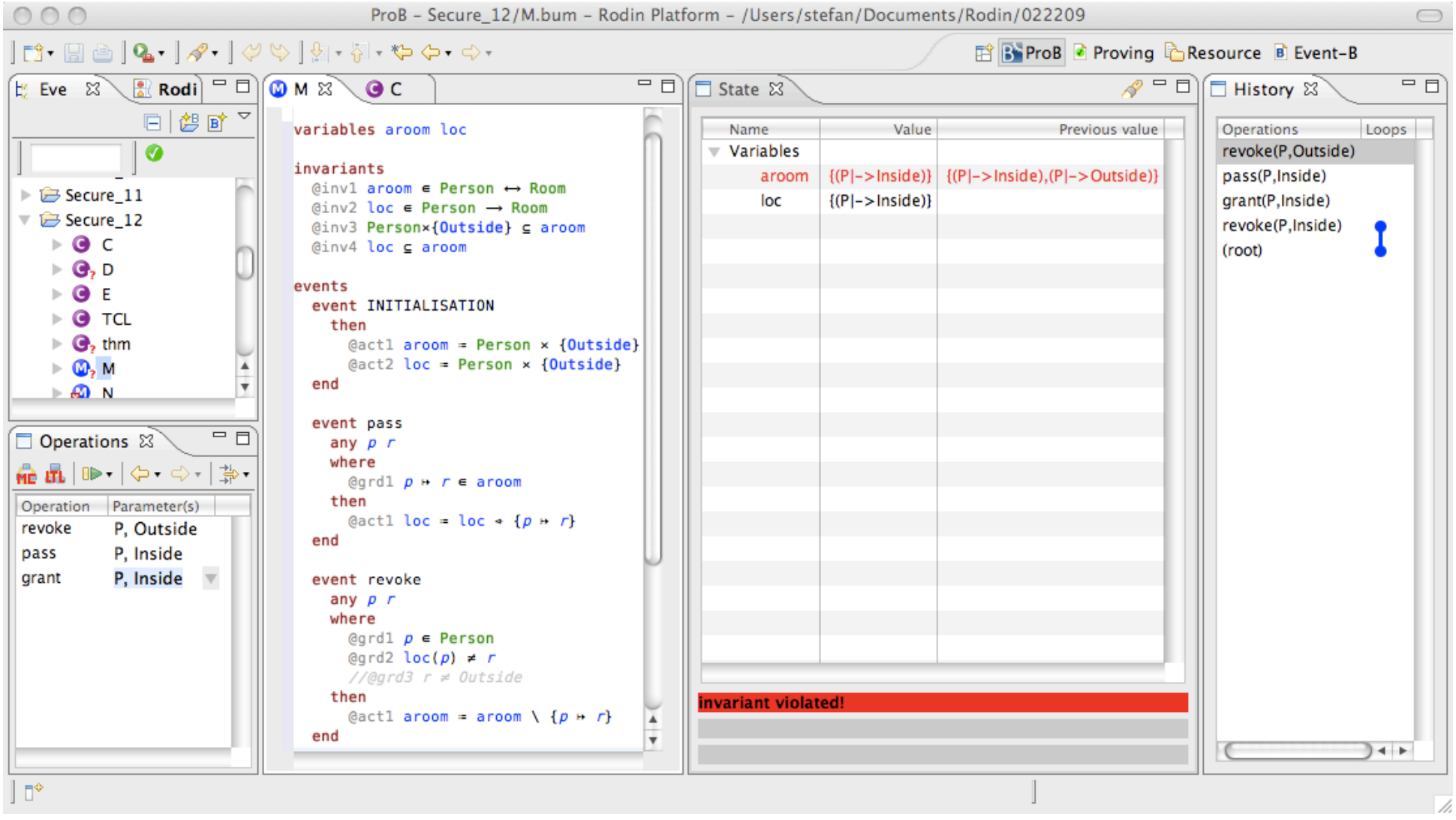
Change of perspective

- Look at a problematic state trace
  (leading to an inconsistent state)



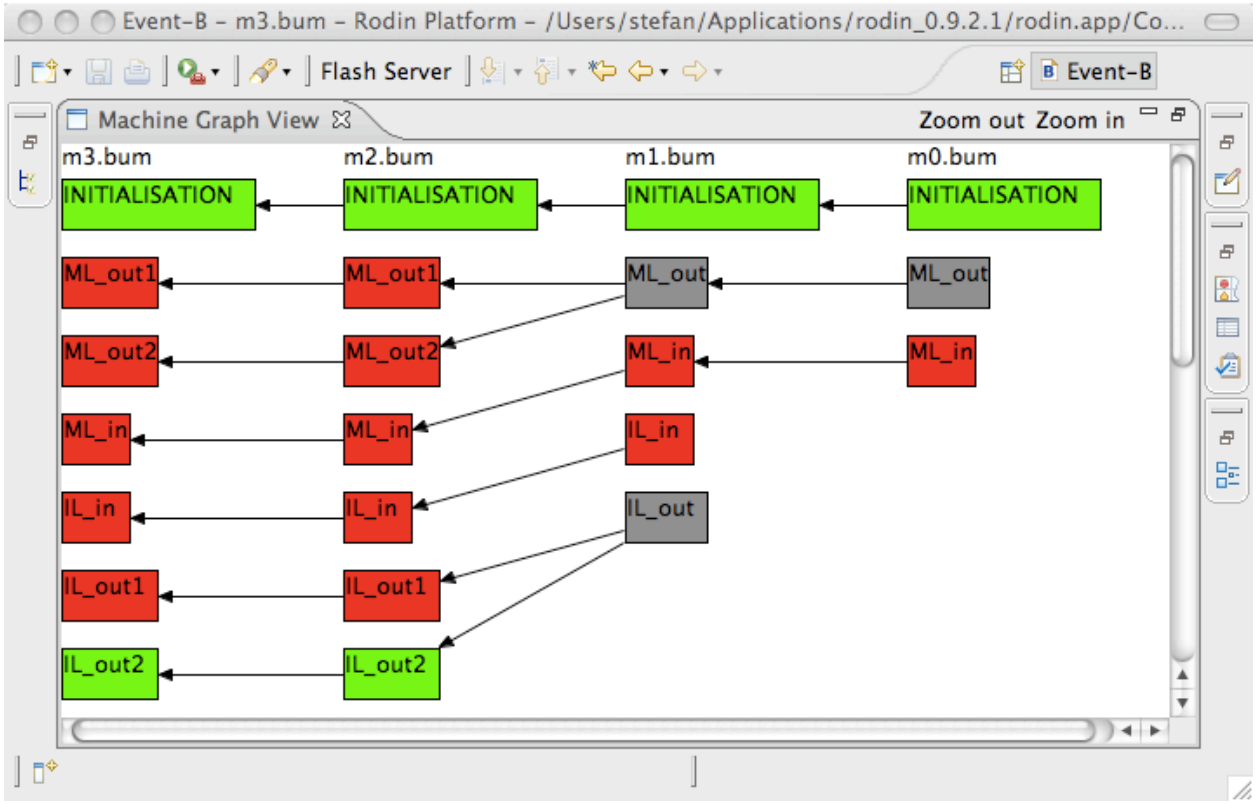
- ProB alerts us that it violates invariant $inv2$:
  $Person \times \{\mathbf{O}\} \subseteq arm$

How the counter example looks in ProB



Contents

Refinement Animation



We work on refinement animation similar to Brama

Refinement Animation

CoffeeR2	*Inv*	*CoffeeR1*	*Inv*	*CoffeeM*
INITIALSATION		*INITIALSATION*		*INITIALSATION*
fill_mug ⟲		*fill_mug* ⟲		*fill_mug* ⟲
drink ↓		*drink* ↓		*drink* ↓
insert_coin ↓		*insert_coin* →		

Sketch of non-graphical display of state of refinement animation

Conclusion

- Teach formal modelling how it is done
- Teach incremental modelling
- Teach how to improve a model in small increments
- Teach making mistakes (how to profit from making mistakes)
- Teach how to explain mistakes and to justify improvements
- Use a software tool like Rodin/ProB in class and in exercises
- Getting a model right is not easy