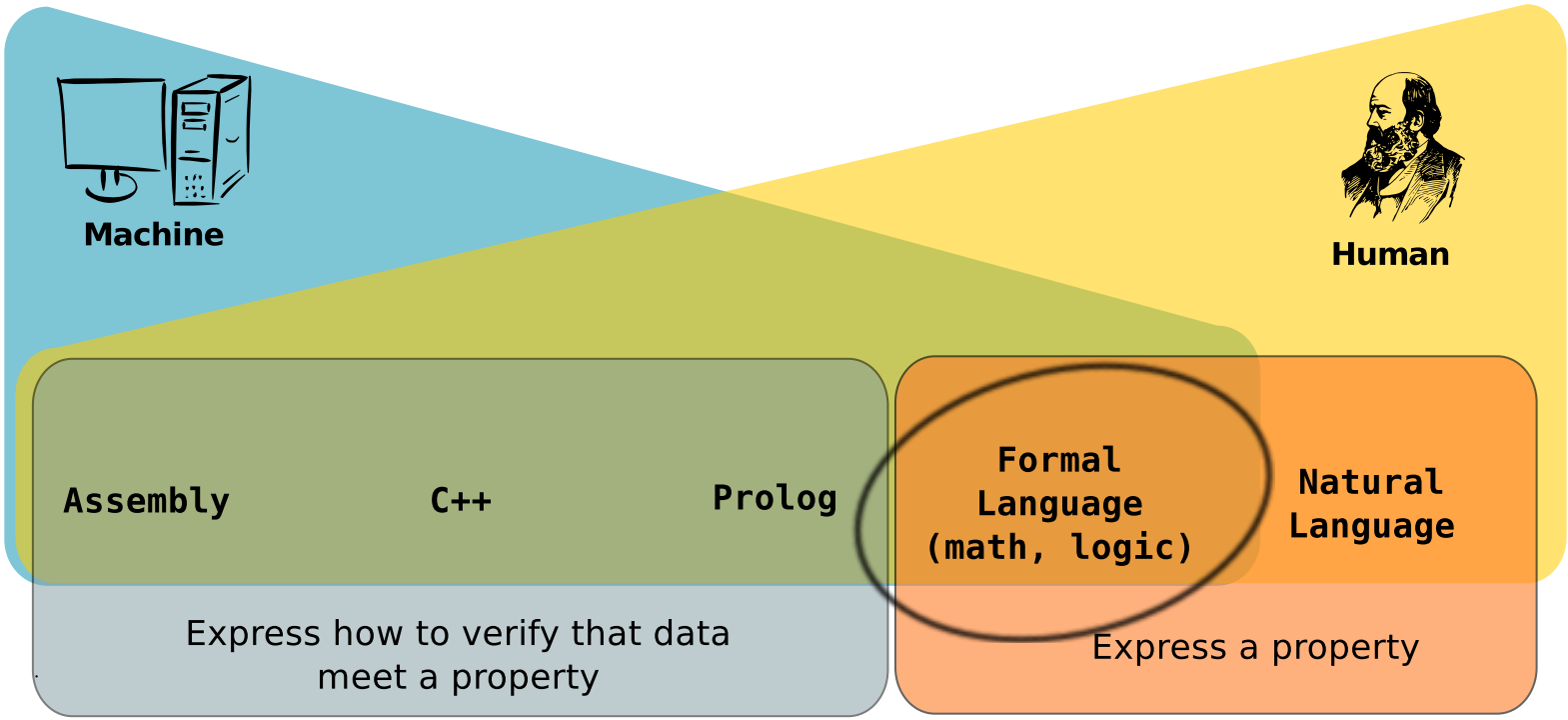
Two B methods : Classical B and Event B

Marielle Petit-Doche

Systerel

April 16th, 2013

Why a formal language ?



Best trade-off between human and machine for expression of properties

Two B approaches

- Both languages developed by Jean-Raymond Abrial
- *Classical B* (in the late 1980s) :
  how the system works
  $\Rightarrow$ functionnal description of a software
- *Event B* (in the late 1990s) :
  why the system works
  $\Rightarrow$ system safety requirements
- Same basic notation based on first order logic and set theory

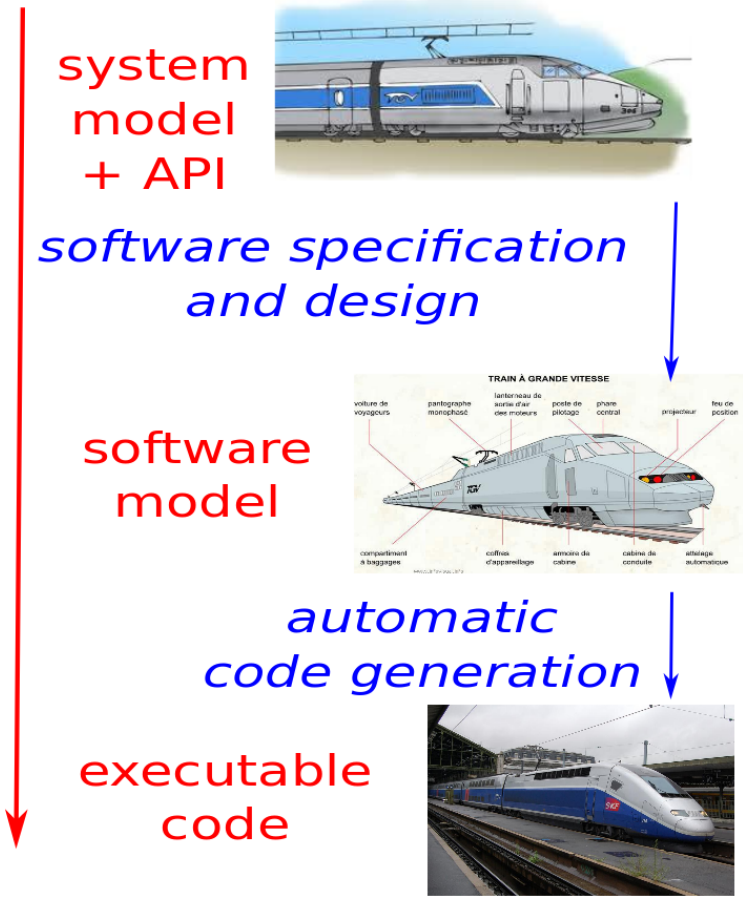
Classical B

Main usage in railway industry

Classical B

Software Design :

- full functional description
- some safety constraints
- structured model
- refinement
- deterministic model to generate code (translation)



Historical

- *The B Book, Assigning Programs to Meanings* [Abrial1996]
- More than 20 years of practice in industry
- First important system : METEOR in 1998
- Used by main manufacturers : ALSTOM, SIEMENS, AREVA,...
- Lots of application in urban railway domain : CBTC, ZC, CC, PMI,..
- Some applications in mainline domain : KVB, Eurobalise,...
- Well adapted to develop critical software according to EN50128

Classical B approach

- Language : first order logic + set theory (inherited from Z notation, Hoare logic,...)
- Structured models : Modules to describe the software architecture
- Operations, Variables, Invariants
- Correct by construction approach
- Formal proof (partially interactive)
- Automatic code translation (C, Ada,...)
- Industrial tool : Atelier B (now free but closed source)

Systerel

Summary

+ Mature approach
+ Well adapted to design critical software : from software informal specification to automatic code generation
+ Easy to maintain
+ Formal verification
+ Industrial approaches and tools adapted to EN50128 requirements

- Methodology and Know-how not shared
- Stable input needed
- Industrial tools are not open source

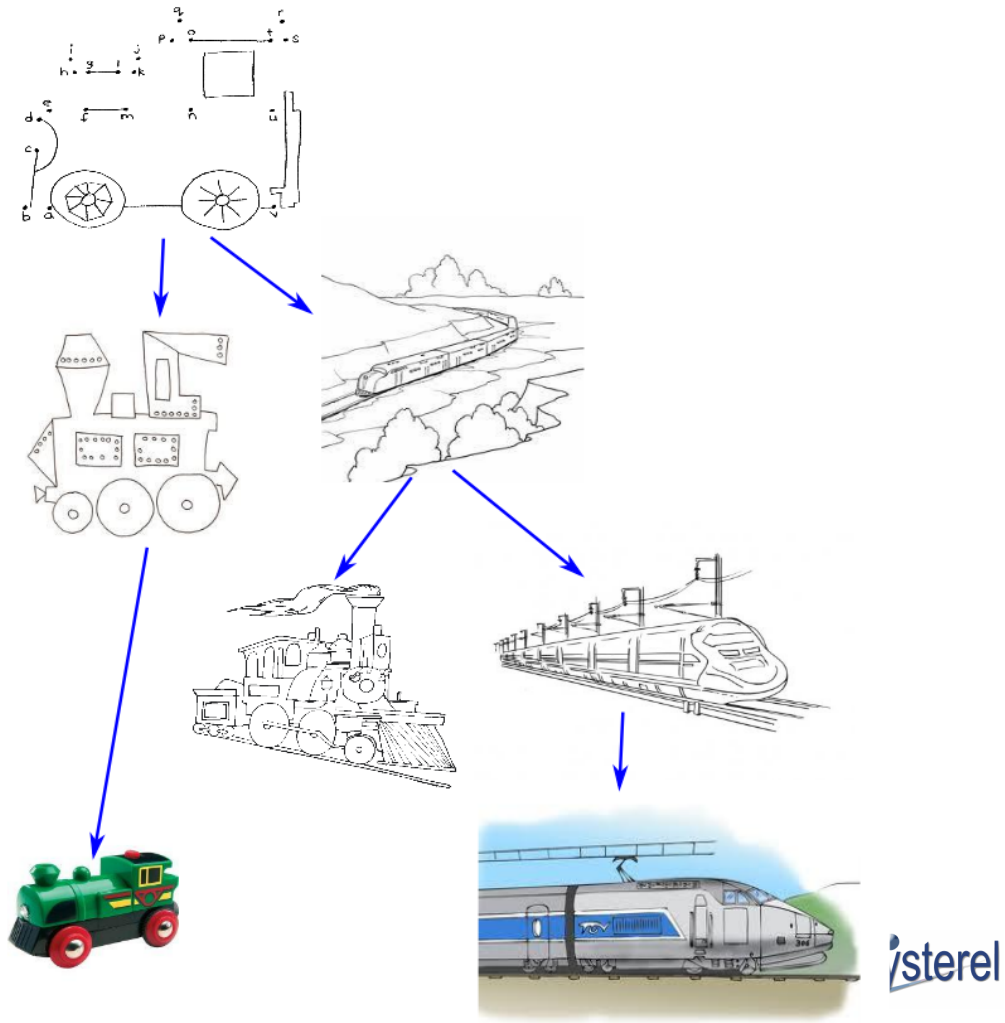
Event B

How to use the event B language

Event B

System analysis :

- initial non deterministic model
- gradual refinements : several levels of abstraction
- proof of safety requirements
- decomposition in sub-systems

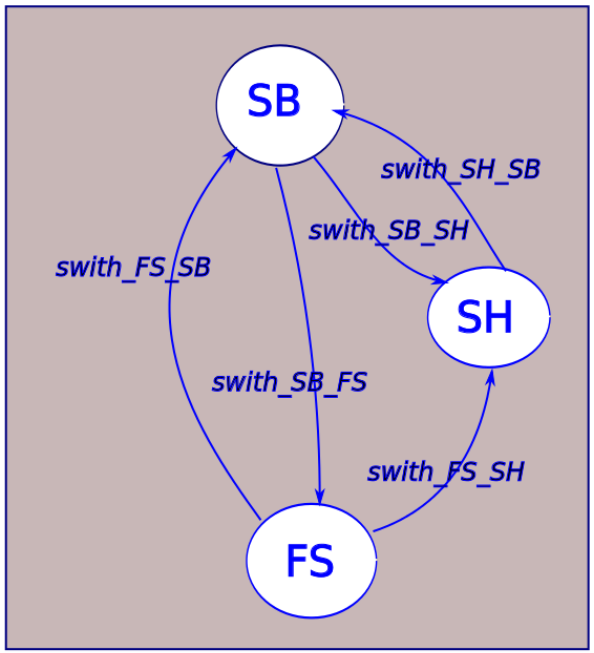


Event B in a quick view

- Seminal article : B Conference 1996
- *Modeling in Event-B* [Abrial2010]
- Same basic formal language as classical B
- Consistency of the model
- Choice of abstraction level
- Formal proof (interactive and automatic)
- Tool : Rodin : open-source tool based on Eclipse
- www.event-b.org

Systerel

Approach : System behaviour via Events



VARIABLES

$State$

INVARIANTS

$State \in \{SB, SH, FS, IS\}$

EVENTS

...

Event $switch\_SB\_SH \widehat{=}$

when

$State = SB$

then

$State := SH$

end

Event $switch\_SB\_FS \widehat{=}$

when

$State = SB$

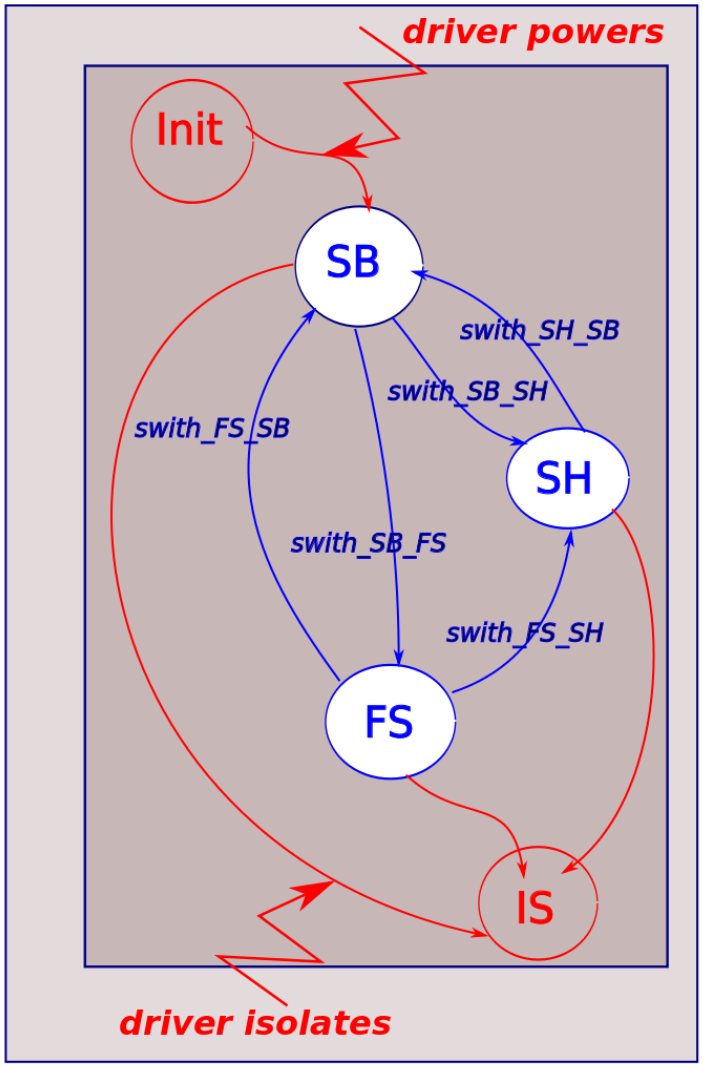
then

$State := FS$

end

...

Closed model : Controller + environment



VARIABLES

$State$

INVARIANTS

$State \in \{SB, SH, FS, IS\}$

EVENTS

Initialisation

begin

$init\_State : State := SB$

end

...

Event $driver\_isolates \widehat{=}$

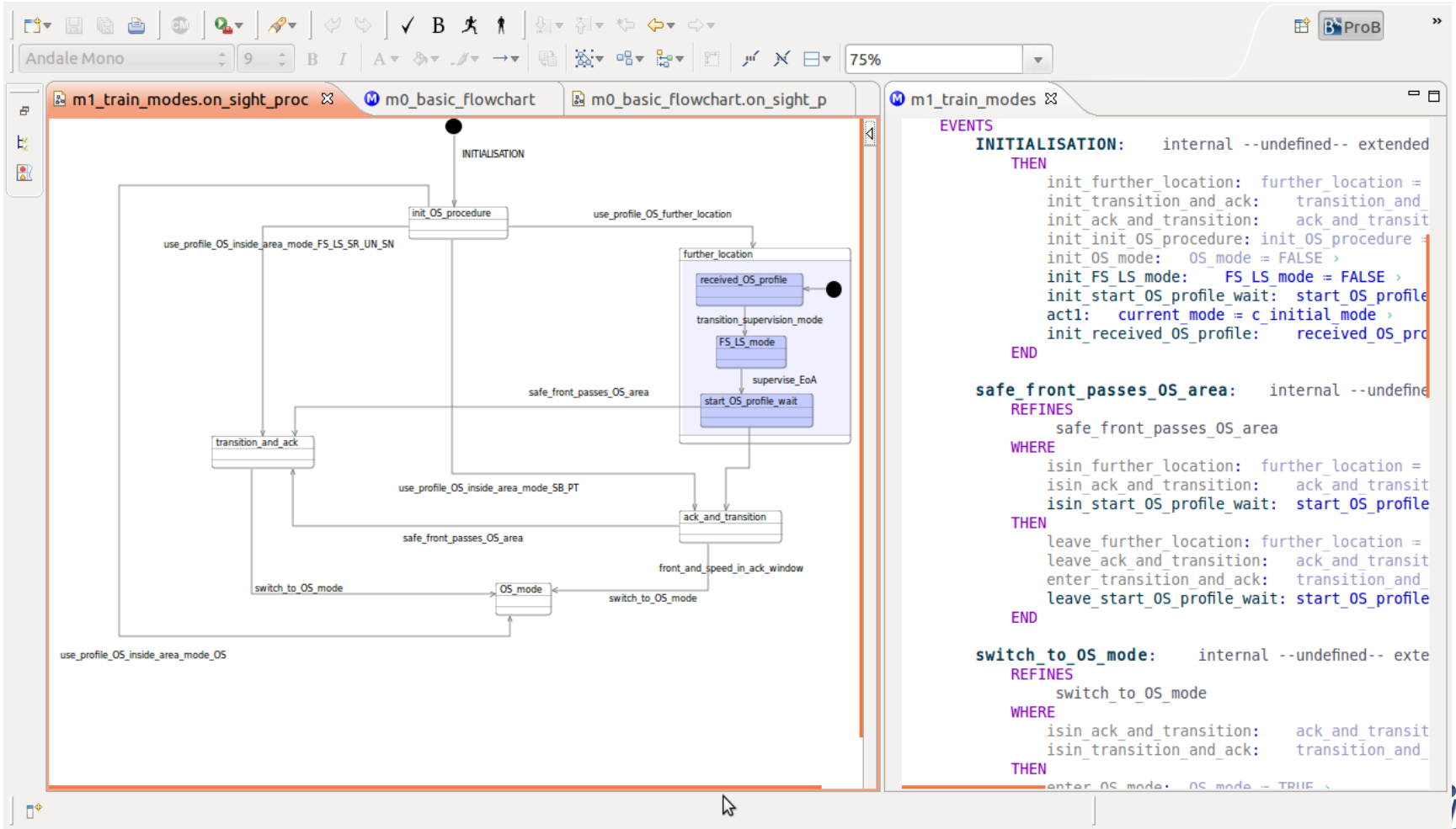
begin

$enter\_IS : State := IS$

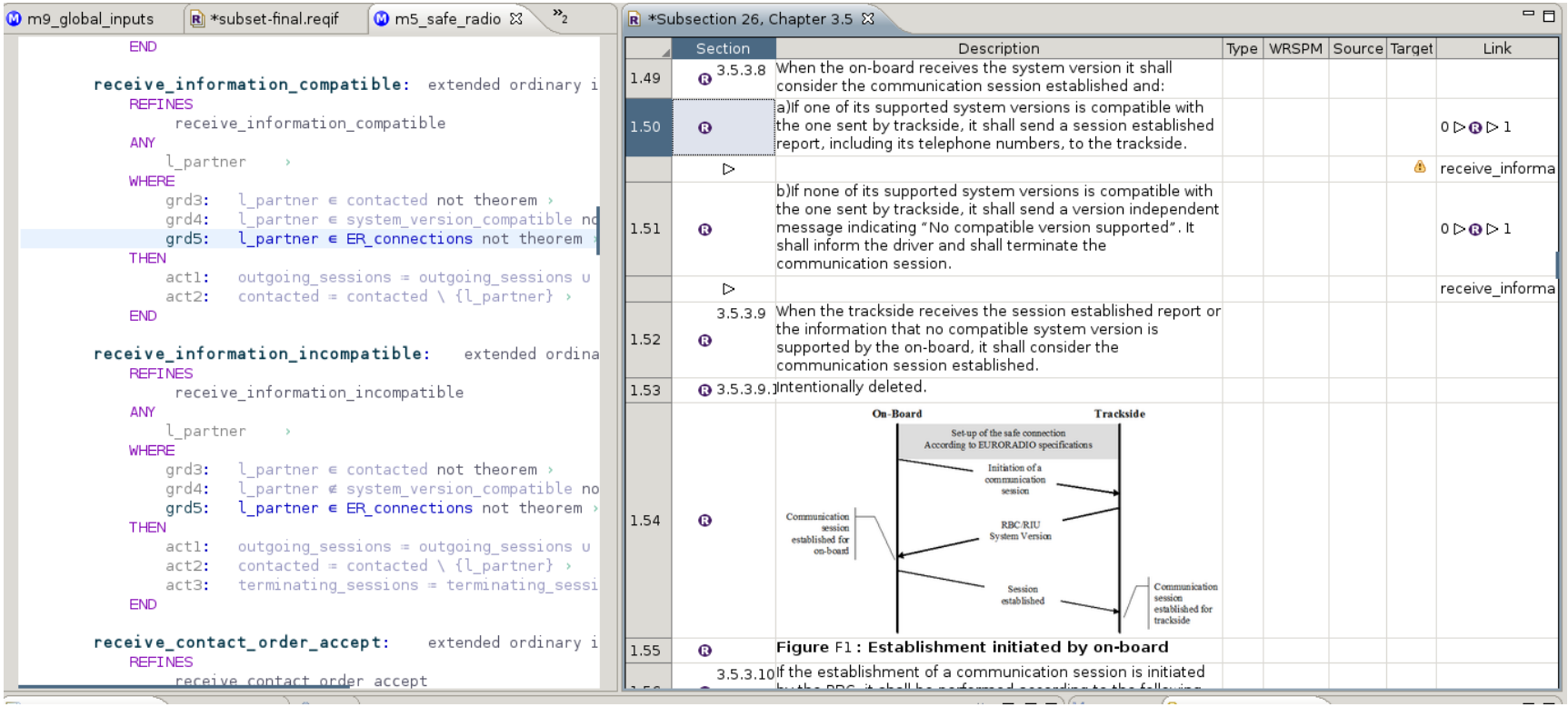
end

END

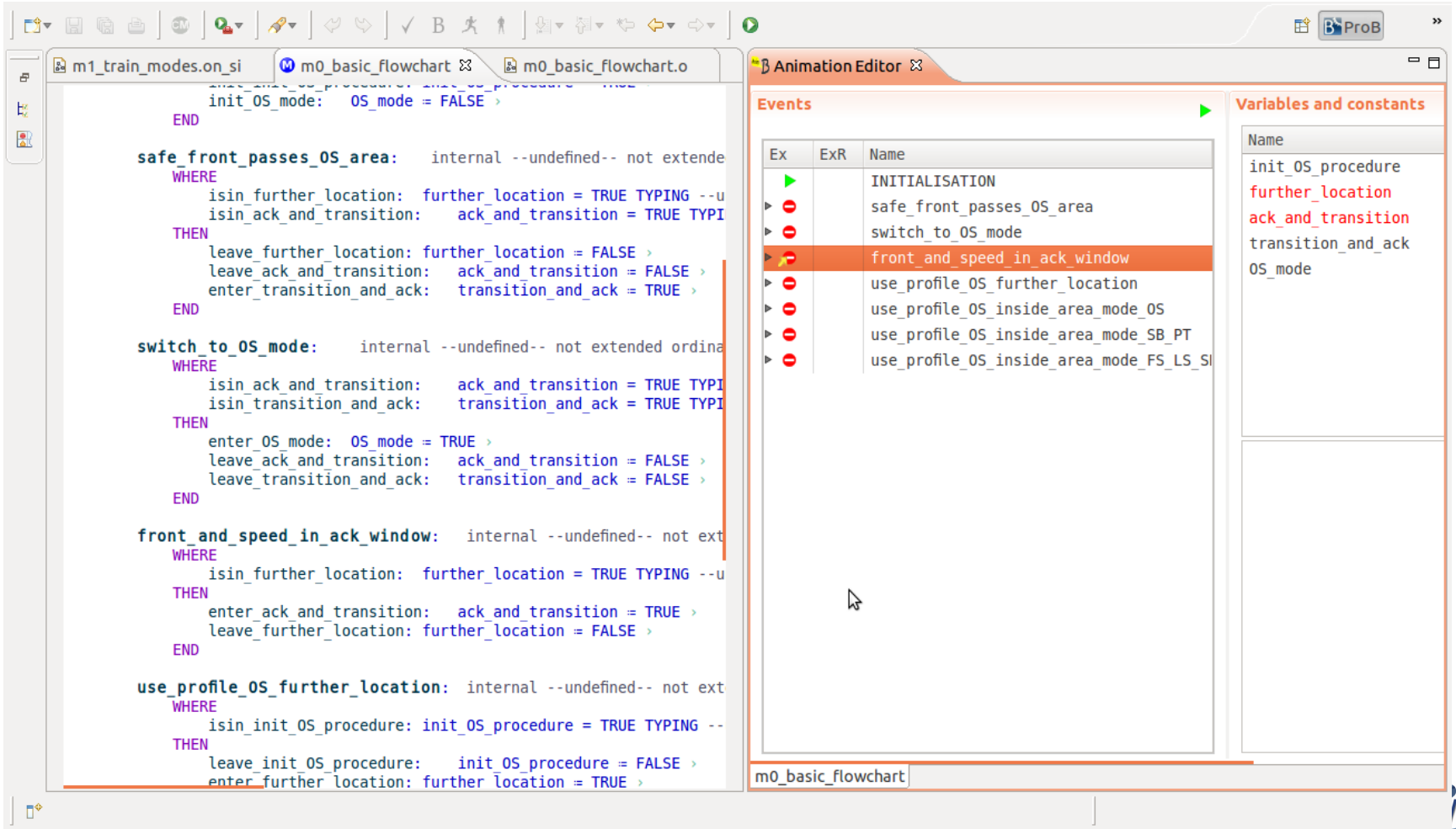
State machine



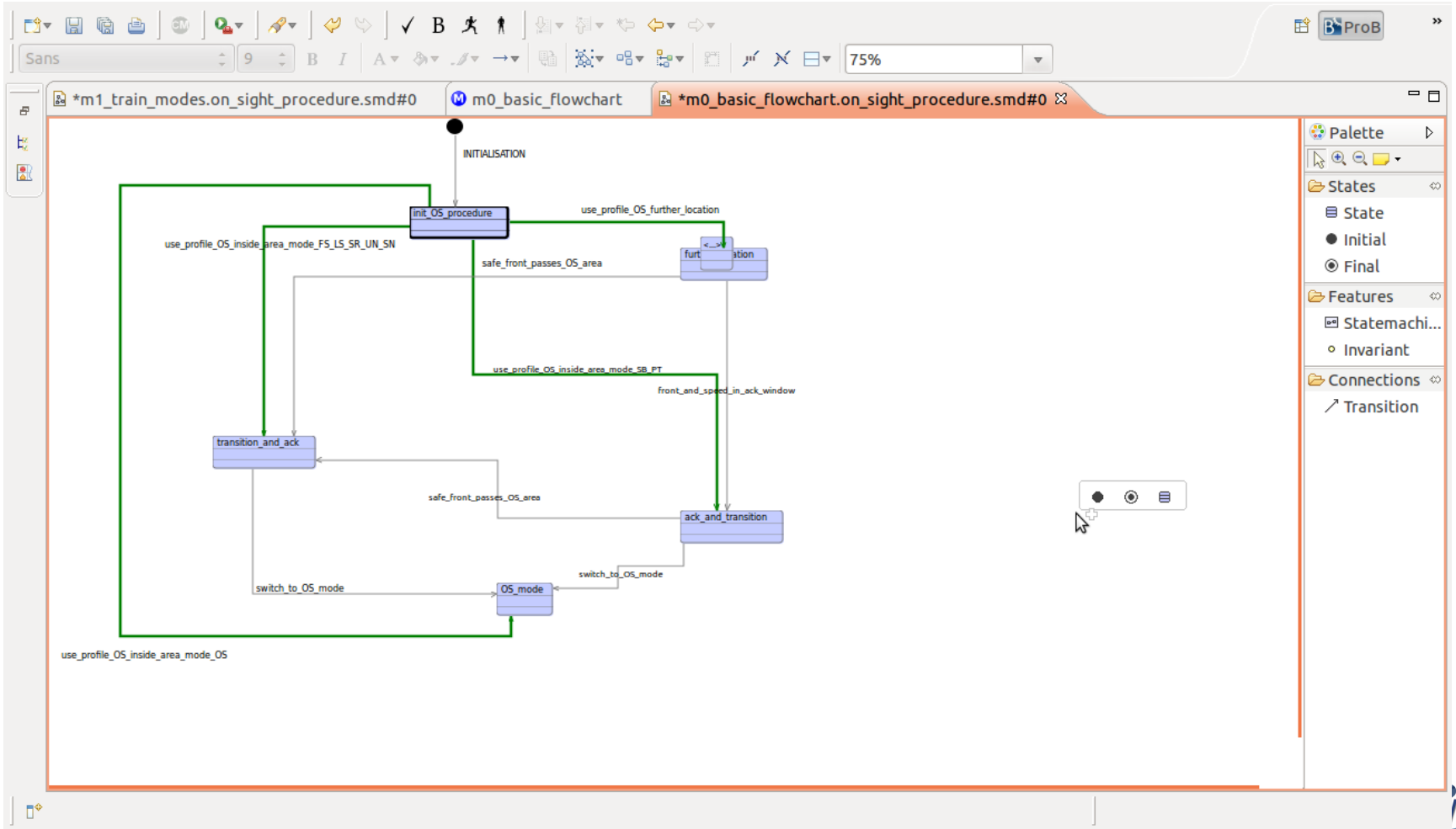
Requirement traceability



VnV : Animation or Simulation (1)



VnV : Animation or Simulation (2)



VnV : Formal Proof : Principle

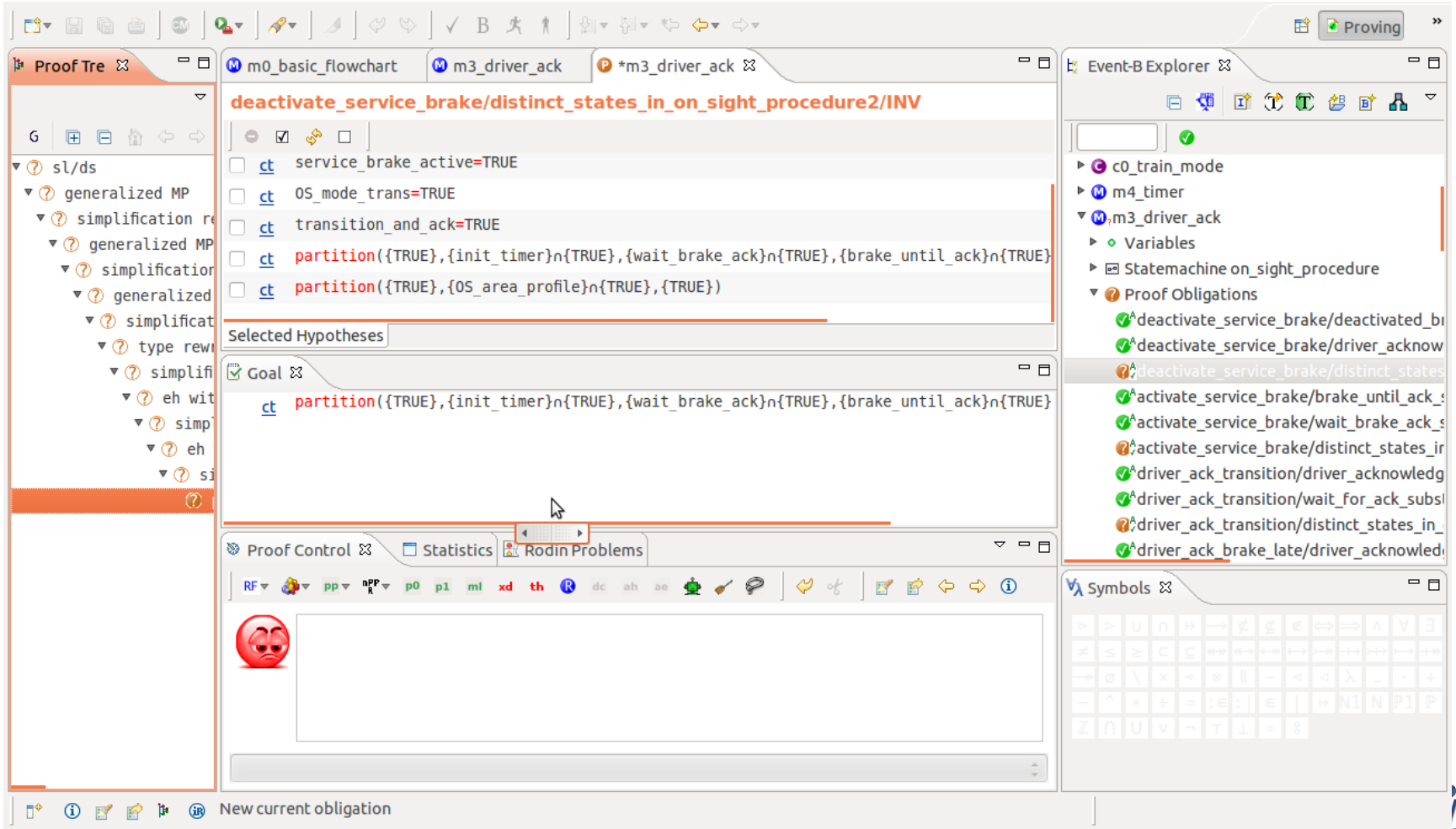
Goal :

- Show the model is sound
- Verify properties

Kind of proof obligations :

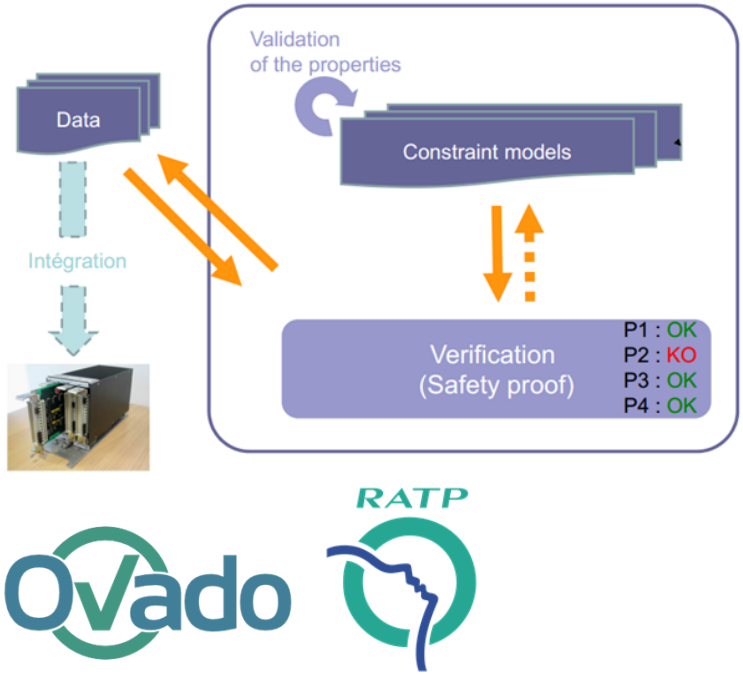
- Invariant preservation
- Non-deterministic action feasibility
- Guard strengthening (refinement)
- Well-definedness
- ...

VnV : Formal Proof : Means



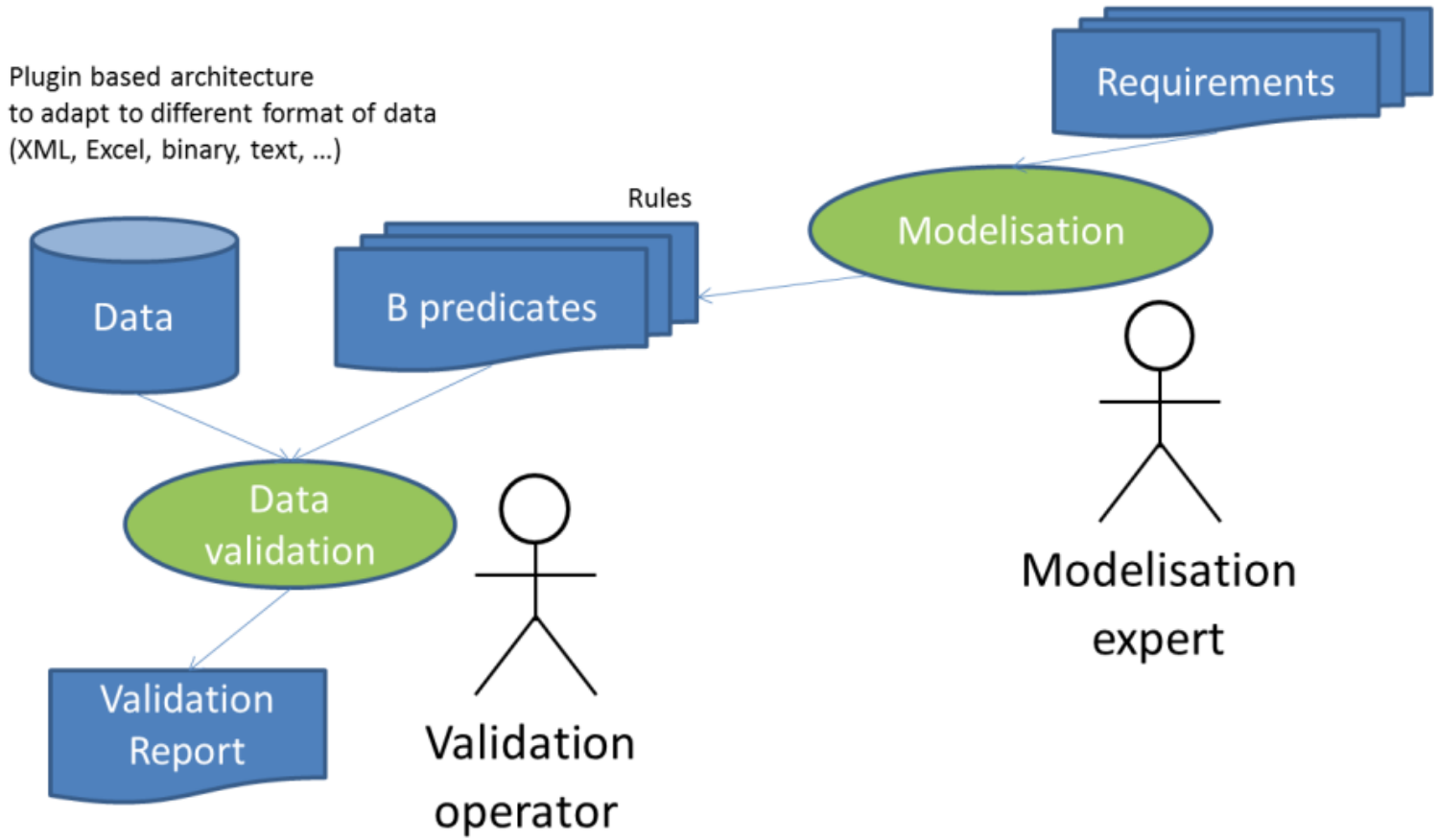
VnV : Data Validation (1)

- Generic systems
- Lots of data to validate
- Separate data preparation from data validation
- Formal model of properties
- Industrial applications
- DS-Event-B [Badeau13]
- ovado.fr



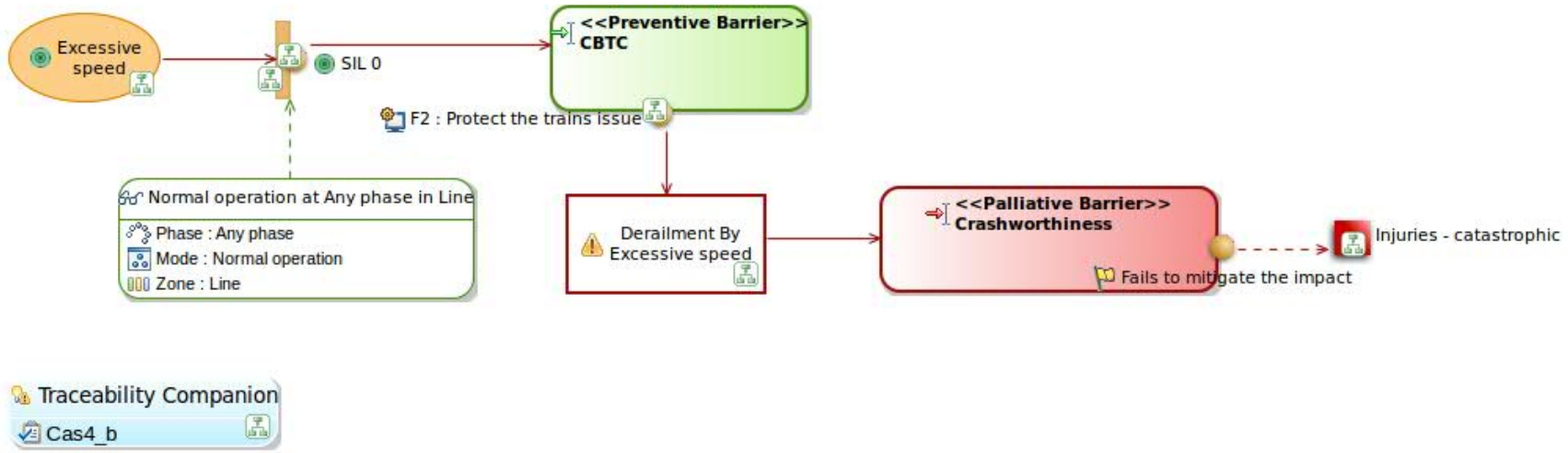
Systerel

VnV : Data Validation (2)

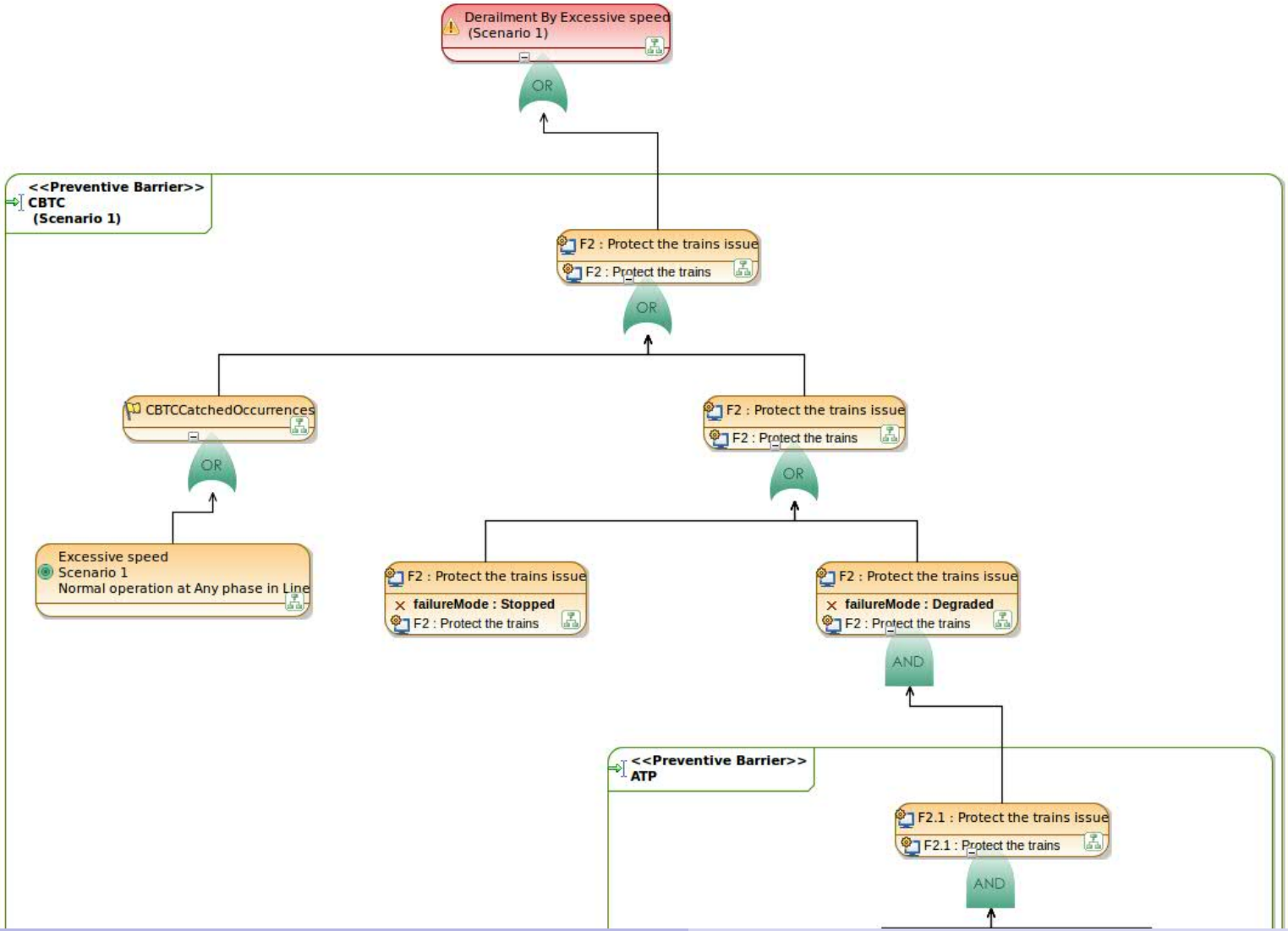


Safety : IMOFIS Safety case

[ERTS2012]

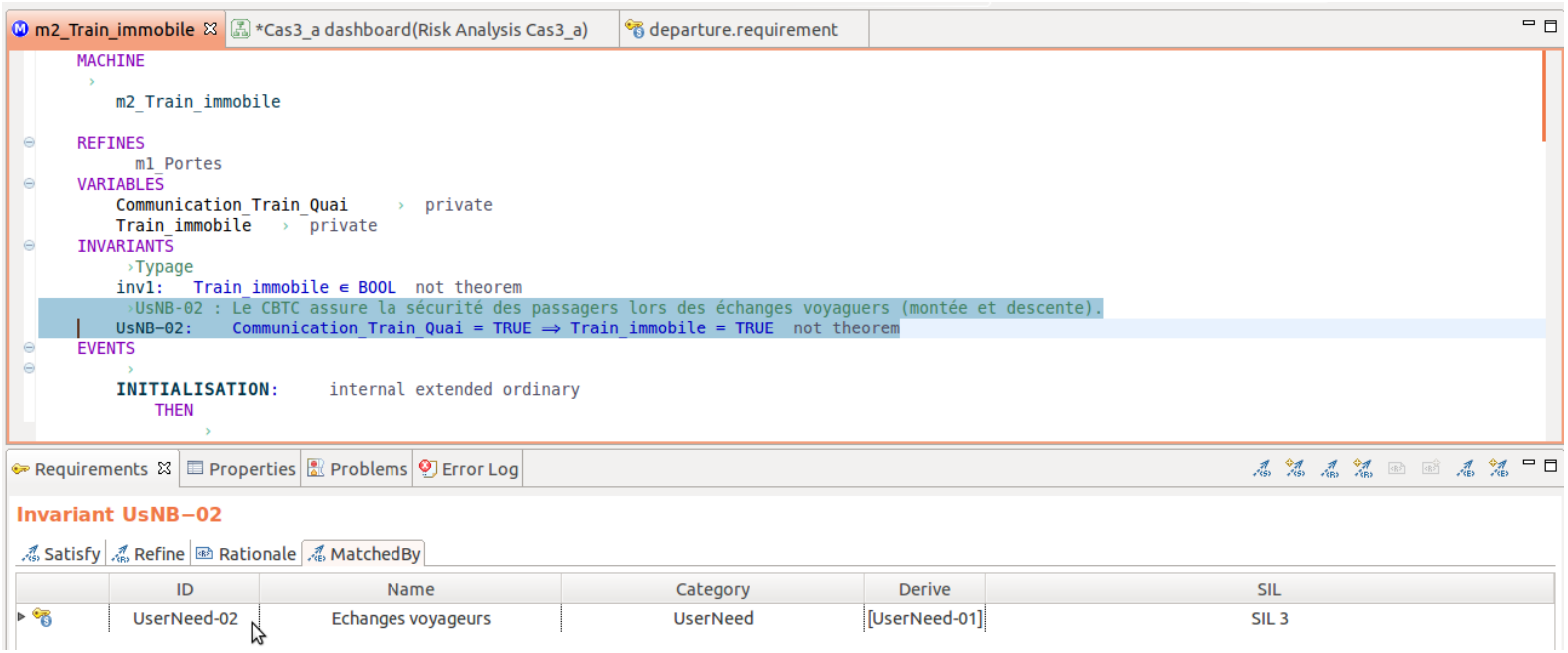


Safety : IMOFIS Safety tree

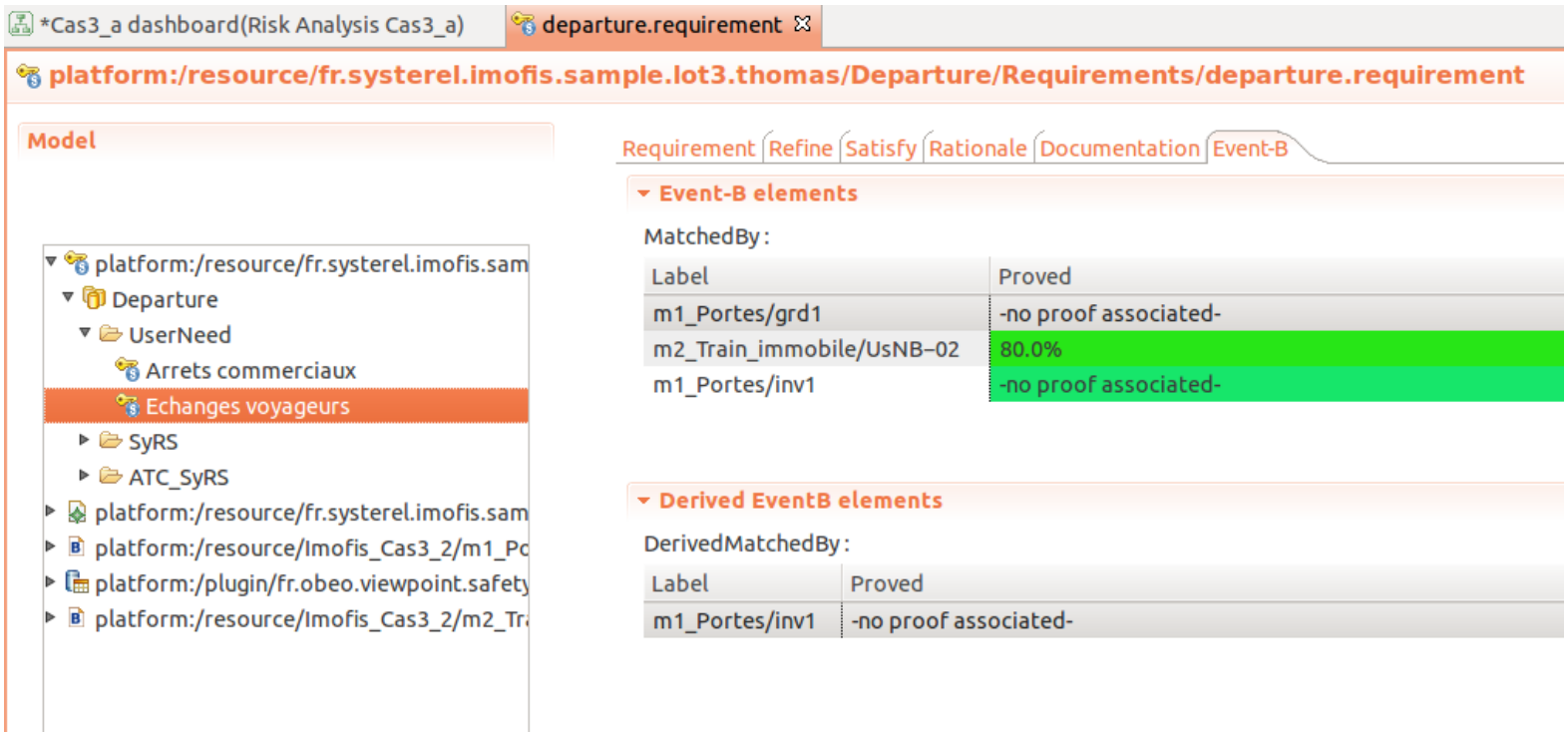


Safety : IMOFIS requirement management

[ERTS2012]

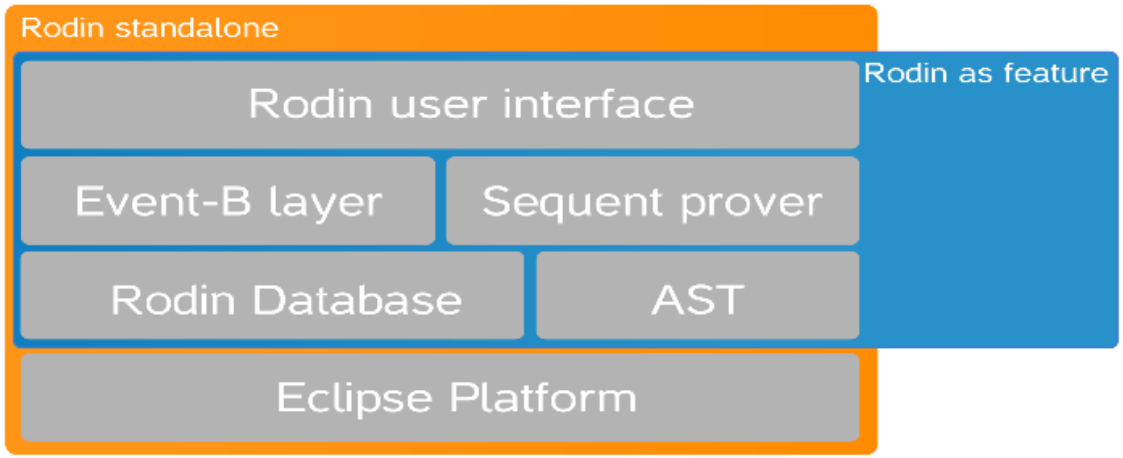


Safety : IMOFIS event B and proof



Rodin platform (1)

www.event-b.org



Rodin platform (2)

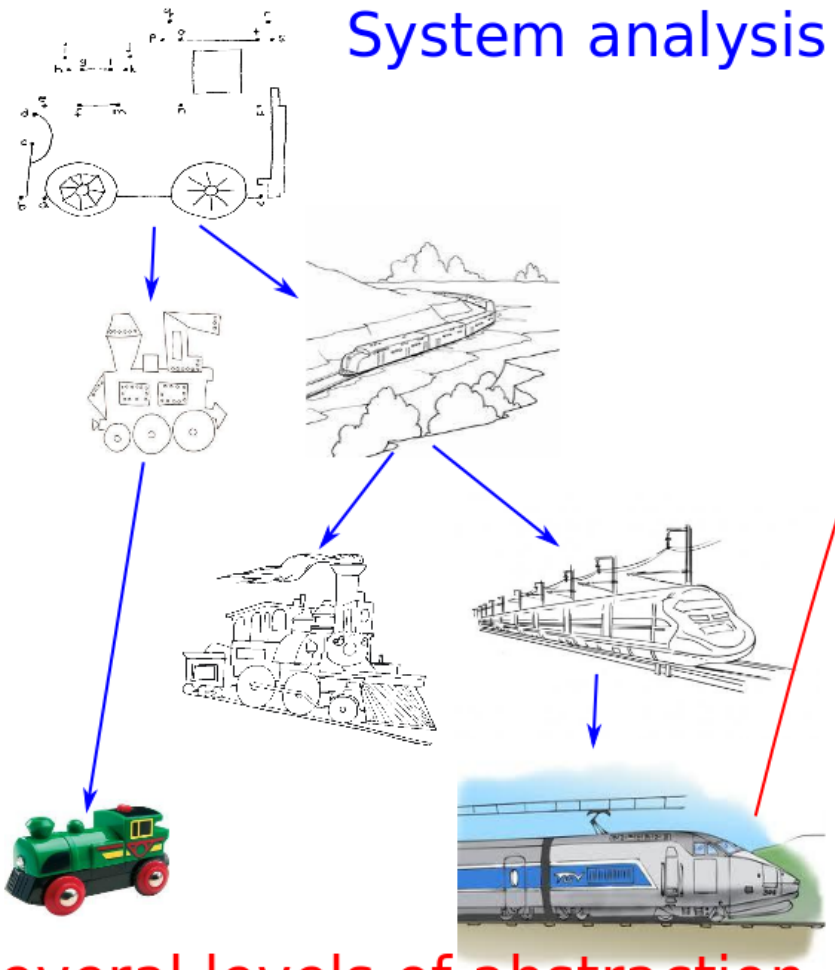
- EPL Licence
- Open approach
- Possibility to add plug-ins :
  - Requirement management : ProR
  - Animation, simulation : ProB, AnimB
  - State machines : IUML
  - Provers : SMT solver, AtelierB
  - Decomposition
  - ...

Summary

+ Well adapted to system analysis
+ Abstraction definition
+ Open source approach
+ Possible links to other (semi)-formal approaches
+ Graphical support for design
+ Simulation of the model
+ Extensible approach and platform
- Little industrial experience
- Not adapted to software design

Event B / Classical B

Event B
System analysis



several levels of abstraction

Classical B
Software Design

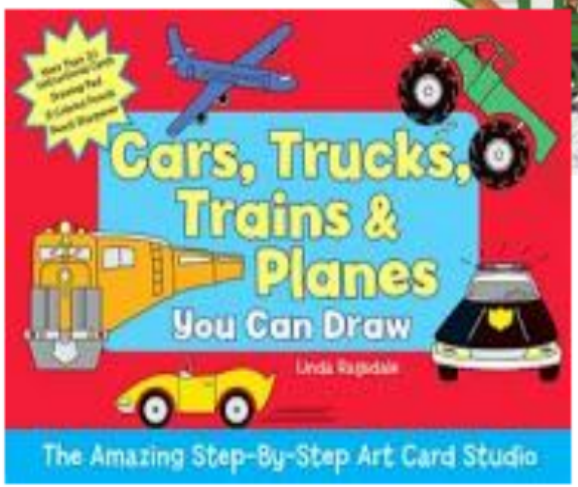
code generation

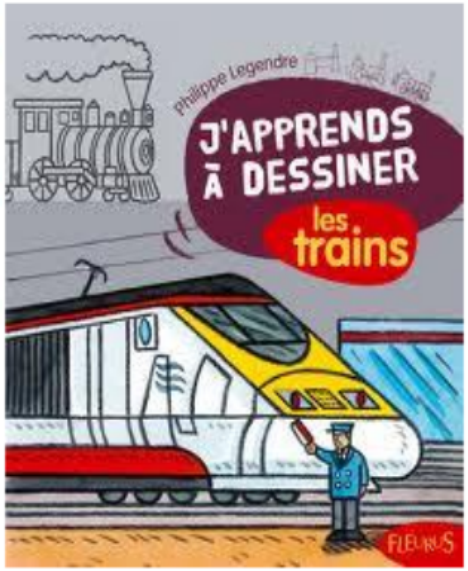
one deterministic model

Guideline definition

Event B
System analysis

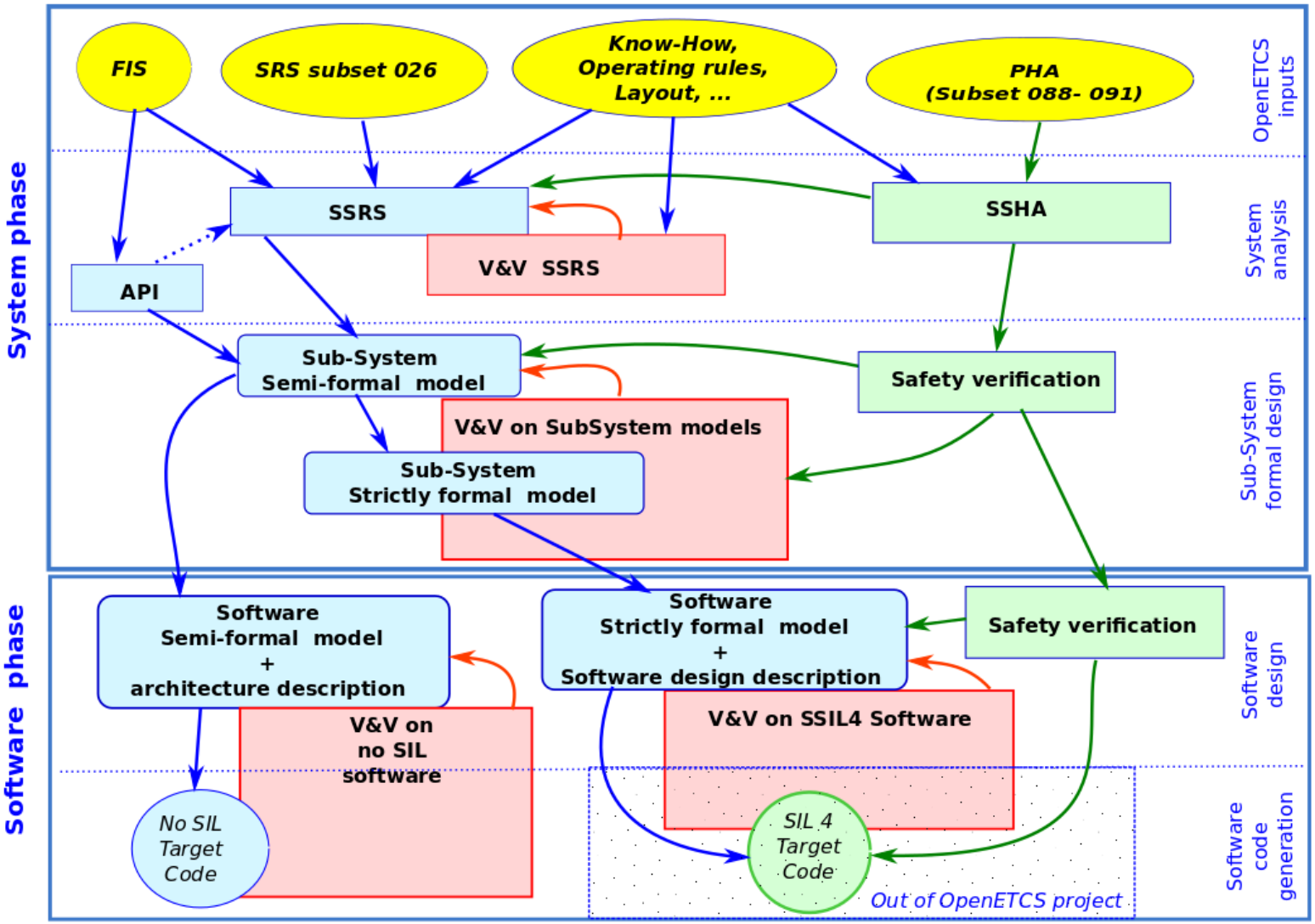
Classical B
Software Design





You need a guideline !

Event B / Classical B in OpenETCS project



Event B / Classical B in OpenETCS project