



# ProB

**Harnessing the Power of Prolog  
to Bring Formal Models and Mathematics to Life**

**Michael Leuschel and the STUPS Team  
University of Düsseldorf**

# What's in a name?



Prolog

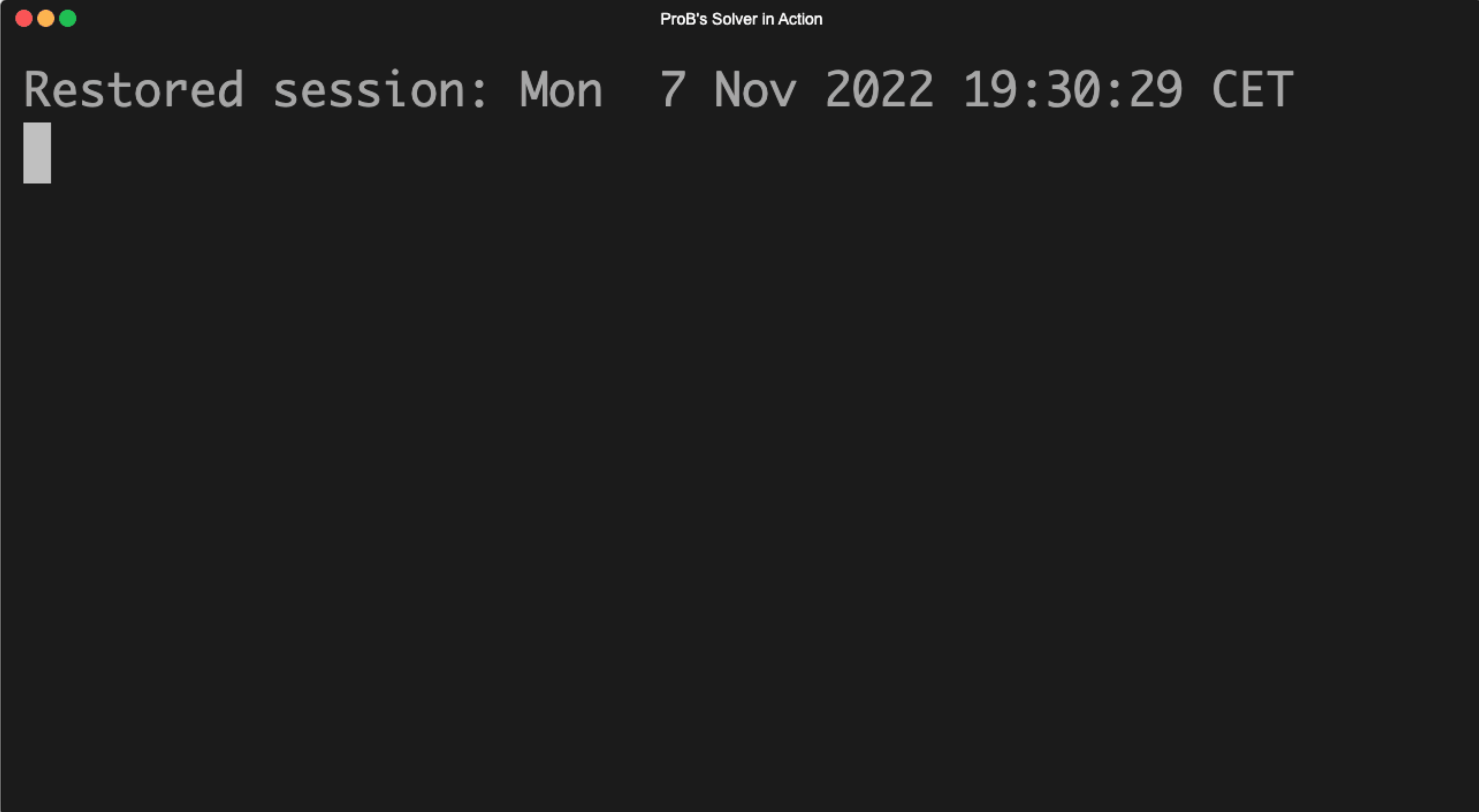
B

A validation tool for the formal method: **B**

**Prolog**

Built on a constraint solver for predicate logic,  
set theory and arithmetic,  
written in SICStus Prolog

# ProB's Solver in Action



```
ProB's Solver in Action
Restored session: Mon  7 Nov 2022 19:30:29 CET
```

The image shows a terminal window with a dark background and light gray text. The window title is "ProB's Solver in Action". The main content of the terminal is the text "Restored session: Mon 7 Nov 2022 19:30:29 CET". There are three colored window control buttons (red, yellow, green) in the top-left corner of the terminal window.

# ProB's Prolog Constraint Solver

## Demo using Jupyter

### Prolog Day 2022



We highlight some of the features of ProB's constraint solving kernel written in Prolog, dealing with unbounded arithmetic, higher-order and infinite sets:

## Some Features

Automatically detecting infinite sets:

**What is the B  
formal method?**



# Formal Methods

- Mathematical techniques to produce correct software and systems: B, TLA+, Z, Alloy, CSP, ...
- Highly recommended for safety critical applications, e.g. for SIL3/SIL4 railway applications by norm EN50128

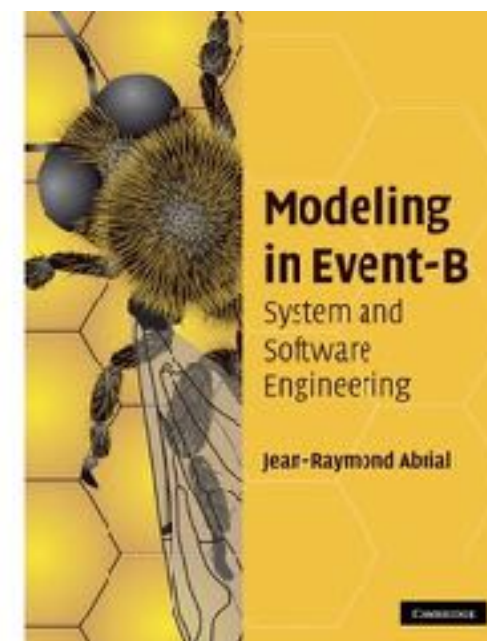




# B Formal Method



**Industrial Applications**



**Mathematical Foundation**



**Tool Support**

# B Formal Method

Line  
14



**B** was used for software for **L14** and **L1**  
**ProB** was used to validate configuration of **L1**

*Paris (AFP)* – Labour unions have called a major one-day strike that threatens to paralyse Paris public transport Thursday in a bid for government action to demand relief from the French government.

The RATP transport operator for the capital has warned of particularly severe disruptions for metro and suburban rail lines, with bus and tram services also impacted by the protest for higher wages.

Seven metro lines will be fully closed and another seven will only operate at peak hours, RATP announced.

Only lines 1 and 14 – which are fully automated with no drivers -- would operate normally but risk becoming overcrowded, the RATP said.

# Origins of B

- Train protection system SACEM for Paris RER Line A, sketch of the B-Method by Jean-Raymond Abrial, 1989 project by Alstom, RATP, SNCF to develop tools and train engineers
- **Paris Metro Line 14** contract won by Matra Transport (now Siemens)
  - **1995:** B tools industrialised by Digilog (now CLEARSY) leading to Atelier-B
  - ready by end 1998: 110 kLOC B model
  - Still in version 1.0, “no single issue caused by software”

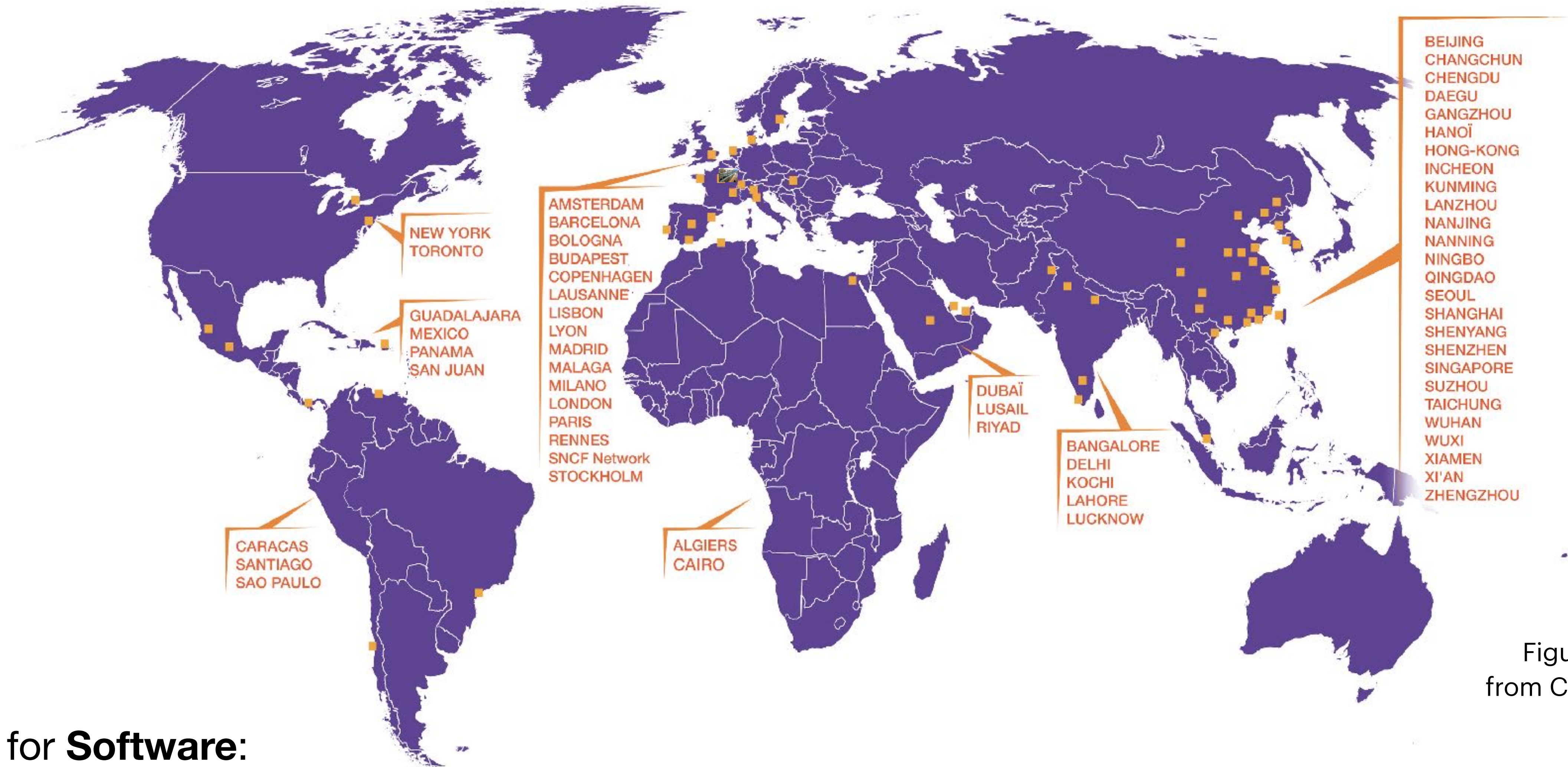


Figure:  
from ClearSy

- **B for Software:**

- about 30% of CBTC systems worldwide employ the B formal method
- Urbalis 400, Alstom, over 100 metro lines worldwide, 25% of worldwide CBTC market



# Validation with ProB



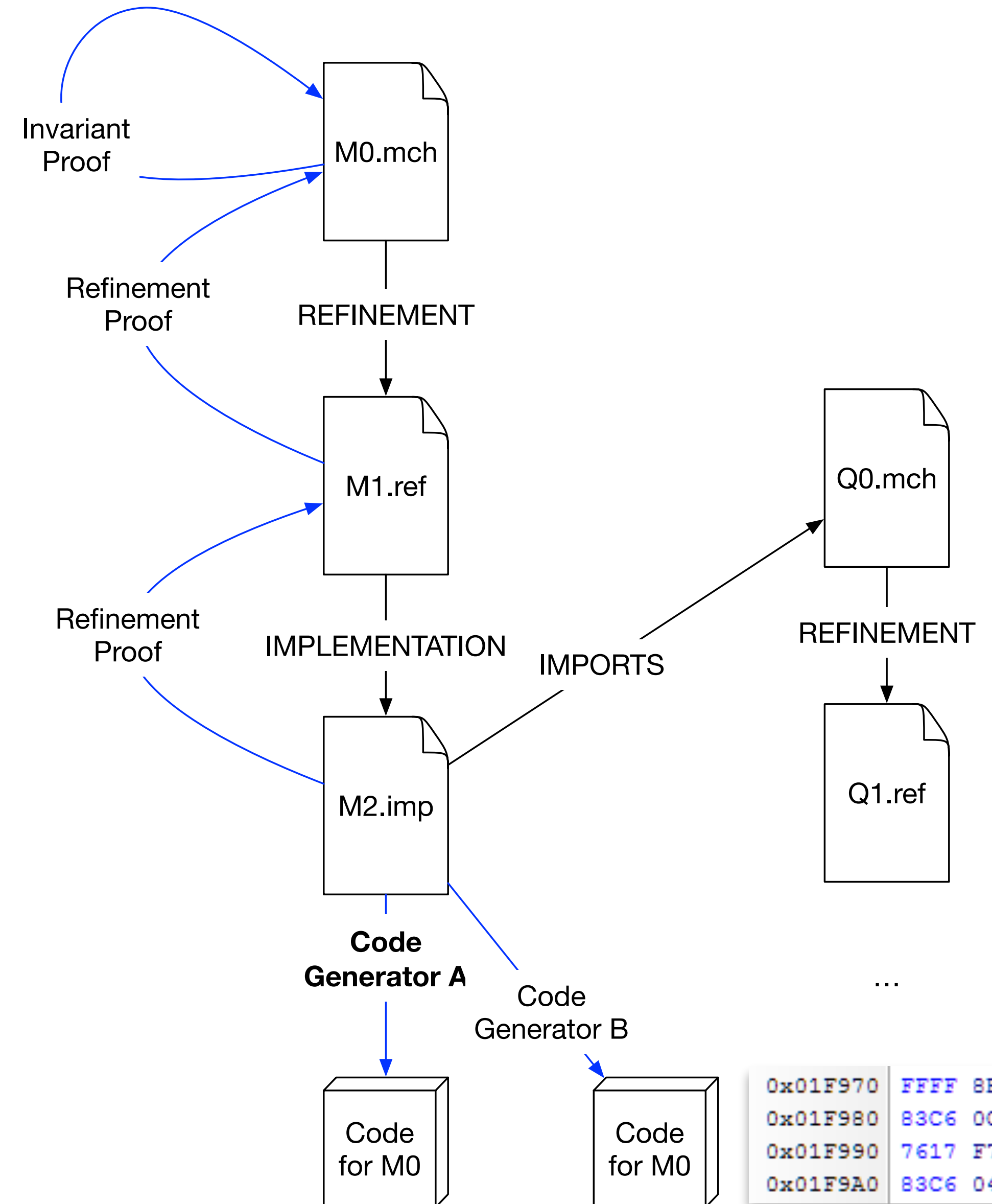
# B Development Process

High-Level Formal Model

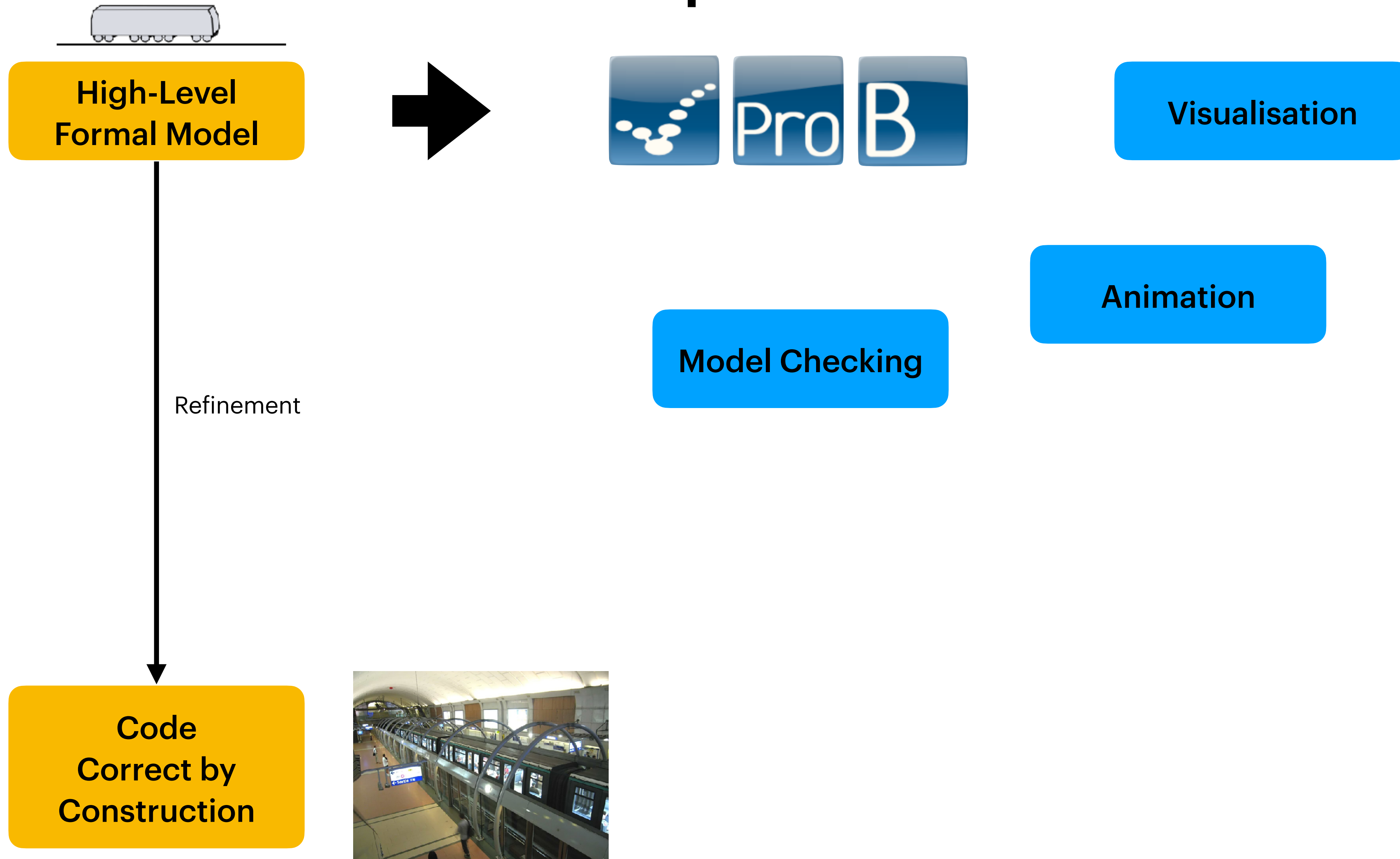


Refinement

Code Correct by Construction



# But who guarantees the correctness of the high-level specification?



# ProB2-UI

The screenshot displays the ProB2-UI interface for the 'SimpleTrainTrack.mch' model. The interface is divided into several panels:

- Operations View:** Located on the left, it shows a list of operations such as 'TTD\_Occupied(ttd=ttd1)', 'TTD\_Occupied(ttd=ttd2)', 'TTD\_Free(ttd=ttd3)', and 'TrainMoveForward'. It includes a 'Filter Operations' search bar and navigation controls.
- State View:** The central panel, titled 'State View | State Visualisation | Edit'. It displays the current state of the model, including variables (e.g., 'occ' with value '{ttd3}', 'train\_rear\_end' with value '29'), constants, sets, invariants, and properties. A 'Filter State' search bar is at the top.
- Project View:** Located on the right, it shows a tree view of the project structure, including models like 'MovingParticles4', 'WasserkocherEinfach\_mch', 'WasserkocherFalsch1\_mch', 'WasserkocherFalsch2\_mch', 'm0\_island\_bridge\_3cars\_mch', 'm1\_bridge\_mch', and 'Lift'.
- Replay View:** Located at the bottom left, it shows a 'Replay' button and a 'Test Case Generation' button. Below it, a table shows the status of the trace: 'SimpleTrainTrack.prob2trace' with a green checkmark.
- Console (REPL):** Located at the bottom center, it shows the 'Interactive Console' with the prompt 'ProB B-Console' and the command 'B> train\_rear\_end' resulting in the value '29'.
- History View:** Located on the right side, below the Project View, it shows a 'History (state 36 of 37)' table with columns for 'Position' and 'Transition'. The transitions listed are 'SETUP\_CONSTANTS', 'INITIALISATION', and multiple instances of 'TTD\_Occupied(ttd=ttd1)' and 'TrainMoveForward'.
- VisB View:** Located at the bottom, it shows an 'SVG-based visualization of current state' of a train track with a train and various colored segments.



# Demo

The screenshot displays the VisB software interface for model checking. The main window is titled "Ref5\_Switch\_mch.eventb - VisB-Examples - ProB 2.0\*".

**Operations Panel (Left):** Lists various operations with red minus icons, indicating they are disabled or not applicable. The selected operation is `con_set_anomaly_output`, marked with a green plus icon. Other operations include `env_start_extending`, `env_extend_gear`, `env_retract_gear`, `env_start_retracting`, `env_start_open_door`, `env_open_door`, `env_close_door`, `env_start_close_door`, `env_open_valve_open_door`, `env_close_valve_open_door`, `env_open_valve_close_door`, `env_close_valve_close_door`, `env_open_valve_retract_gear`, `env_close_valve_retract_gear`, `env_open_valve_extend_gear`, `env_close_valve_extend_gear`, `con_stimulate_open_door_valve`, and `con_stop_stimulate_open_door_valve`.

**State View Panel (Top Center):** Shows the current state variables and their values:

Name	Value
analogical_switch	switch_closed
general_EV	TRUE
general_valve	valve_open
handle_move	TRUE
handle	up
last_handle_state	{up}
lock_door_opening	FALSE
shock_absorber	flight

**Model Checking Results (Right):** Shows the status of the model checking process. The "Status" column indicates success (green checkmarks). The "Description" column provides details about the checks performed, such as "Mixed BF/DF, Deadlock check, Invariant check" and "Mixed BF/DF, Find other errors, Additional go...". The "Message" column shows "Goal found".

**Time elapsed: 0.1 s**

**Progress: [Progress Bar]**

**History (state 13 of 13):** Shows the sequence of transitions executed during the model checking process:

Position	Transition
0	---root---
1	INITIALISATION
2	env_toggle_handle
3	con_stimulate_general_valve
4	con_stimulate_open_door_valve
5	env_open_valve_open_door
6	env_close_switch
7	envn_open_general_valve
8	env_start_open_door
9	env_open_door
10	con_stimulate_retract_gear_valve

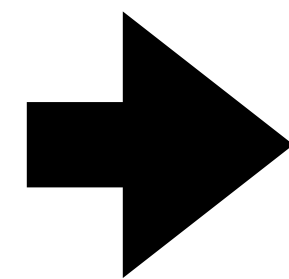
**State Visualisation (Bottom Center):** Shows three sequential diagrams of a mechanical system (a door/gear assembly) illustrating the state changes during the model checking process. The diagrams show the handle moving up, the valve opening, and the door opening.

**Animation Panel (Bottom Left):** Shows the "Replay" button and the "Symbolic" tab selected. The "Test Case Generation" button is also visible.

**Status Bar (Bottom):** Displays "Everything is OK" with an information icon.

# Can we use the high-level specification before a complete implementation?

High-Level Formal Model



Visualisation

Model Checking

Animation

Refinement

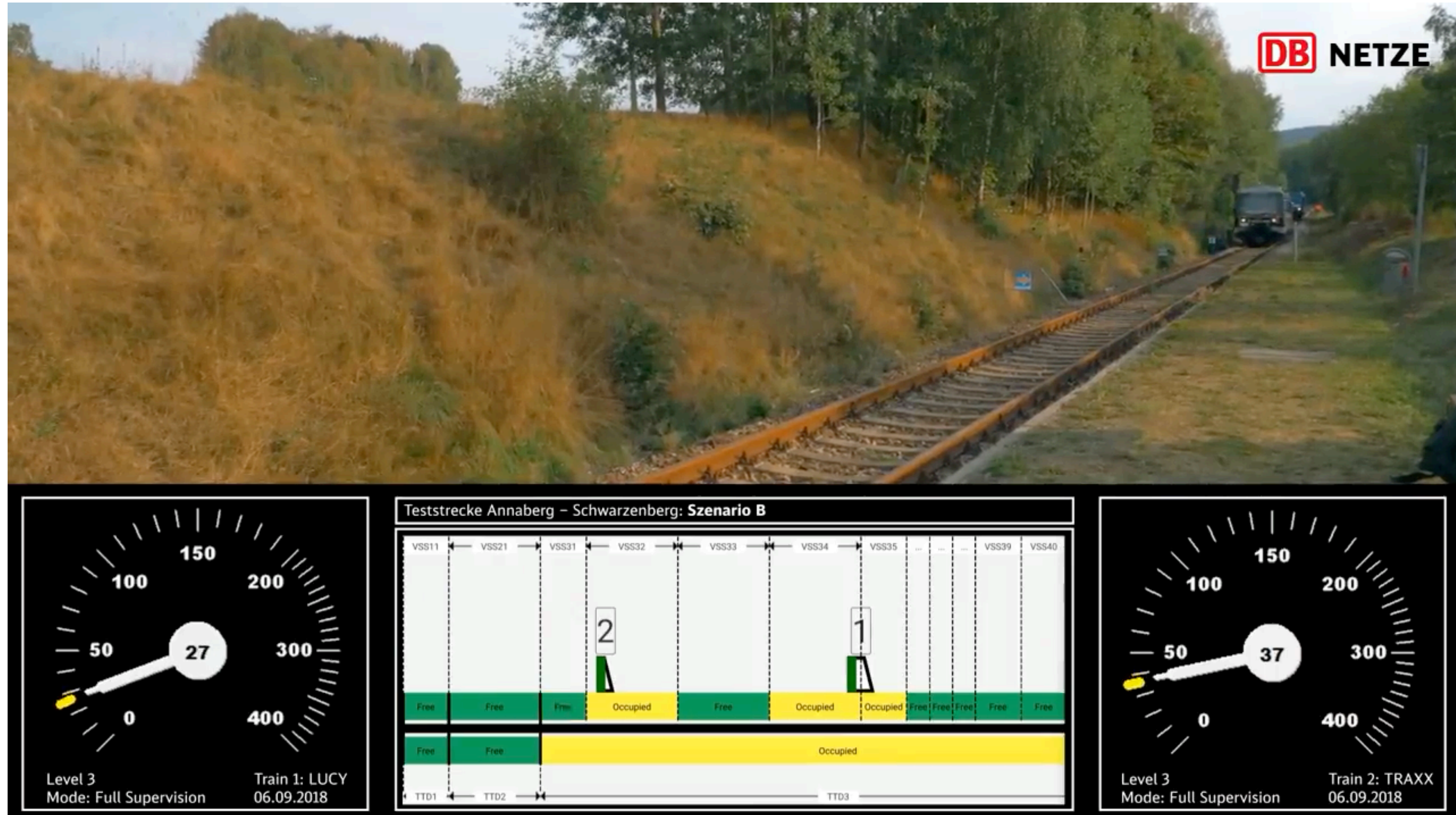


Code Correct by Construction



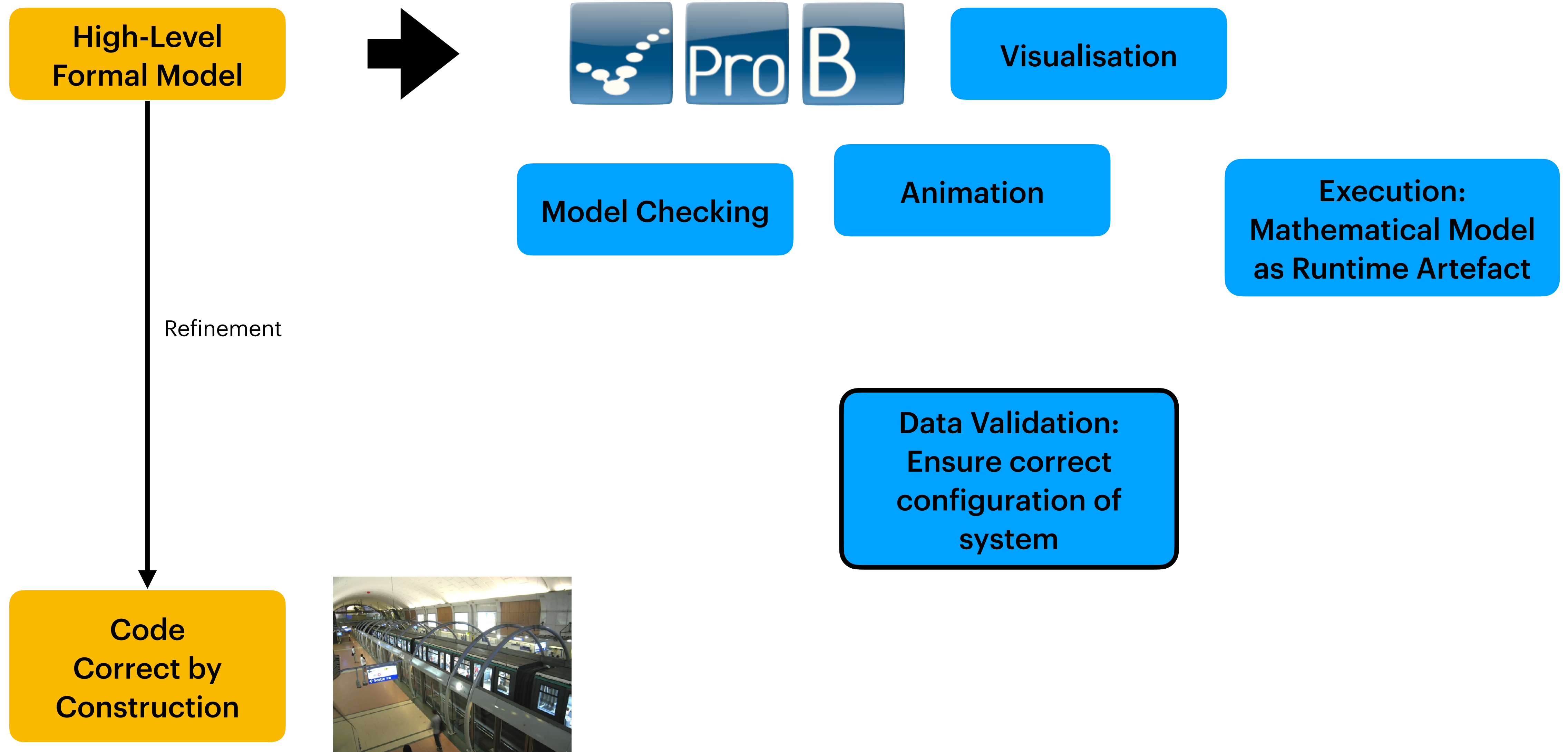
Execution:  
Mathematical Model  
as Runtime Artefact

# ProB running in SICStus Prolog in real-time executing a formal B model of the Hybrid-Level 3 principles



Train 2 following Train 1 (Lucy) on the same occupied track section, but on different virtual subsections

# But who guarantees the correct configuration of the final system?



# Data Validation at Siemens

One man-month of work  
done automatically in 3  
minutes with ProB  
and undetected issues found



# ProB at Siemens

*The work done with ProB is a **great success for Siemens**:*

- Thanks to the automatization and **ProB**, the wayside data validation is **quicker**, more **complete** and **easier** than AtelierB.*
- The on-board data validation, which was not formally proven, is now proven with **ProB**, so the use of formal methods has been extended.*
- The B experts are only required when problems have been found, whereas before highly skilled people were required for long and fastidious B modification and proof.*
- The validation of **ProB** itself enables a use in a **SIL4** development.*



Certified as T2 Tool  
EN50128



Jérôme Falampin, SAS Siemens

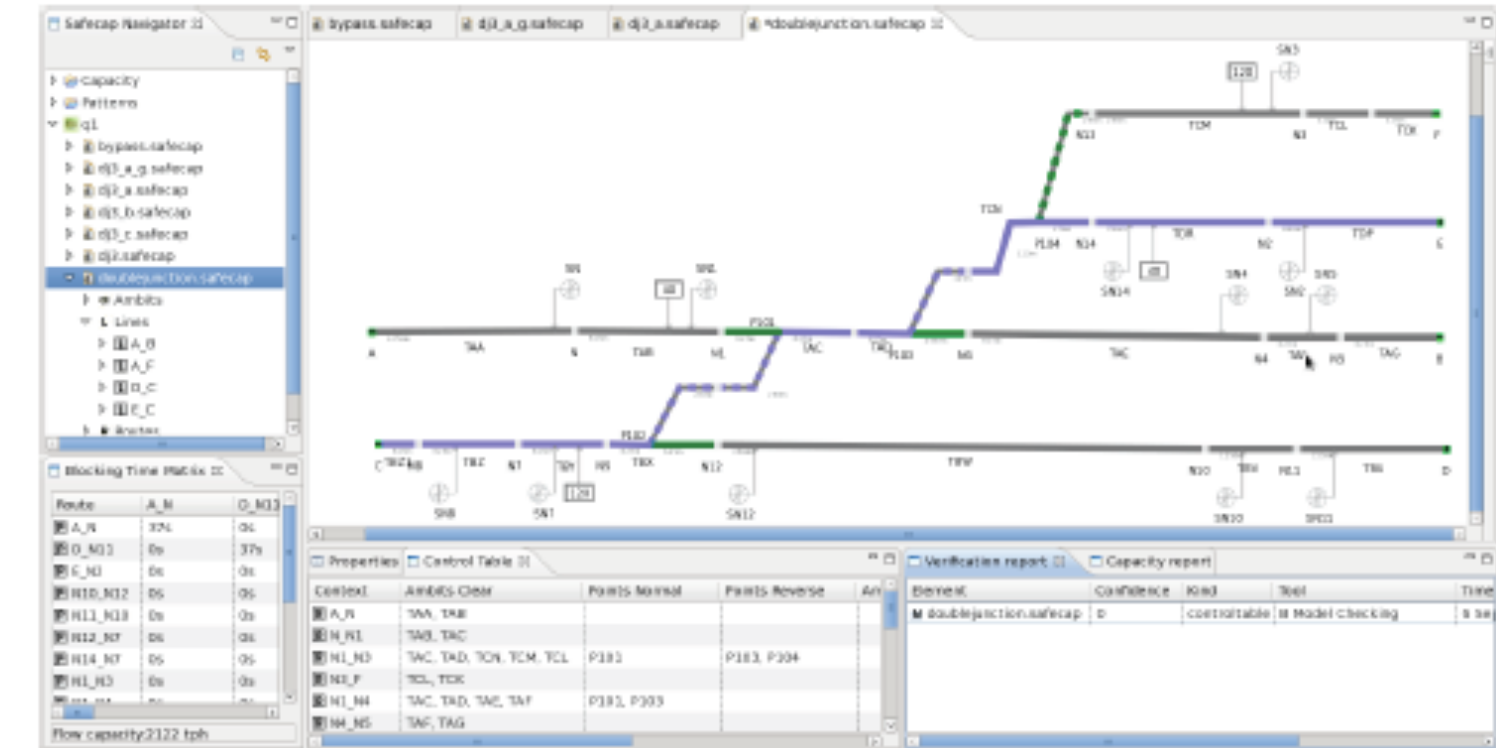
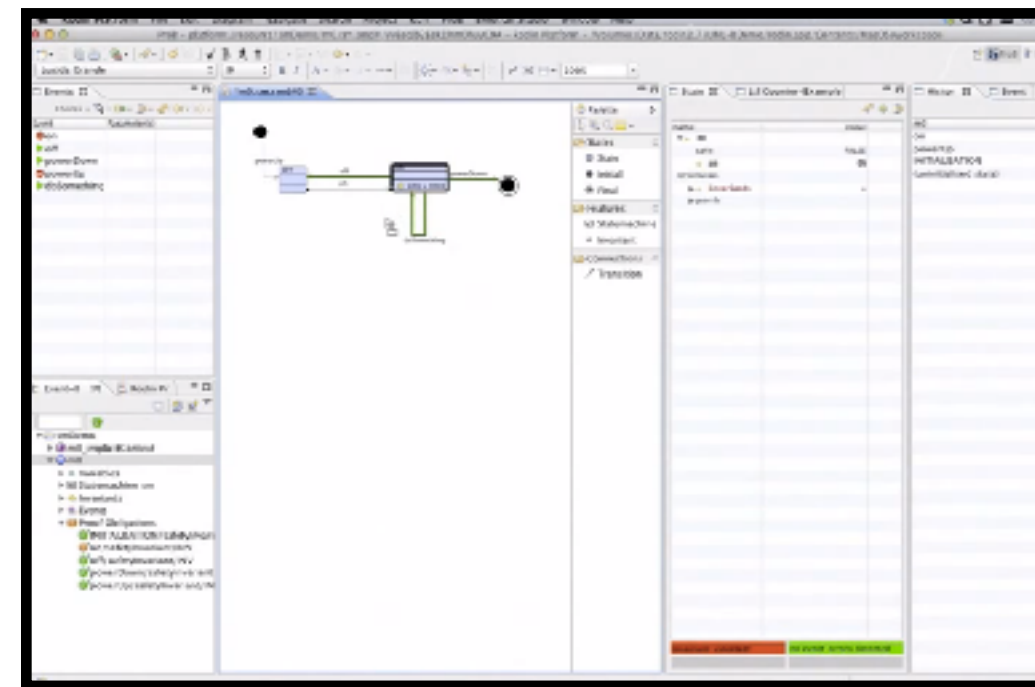
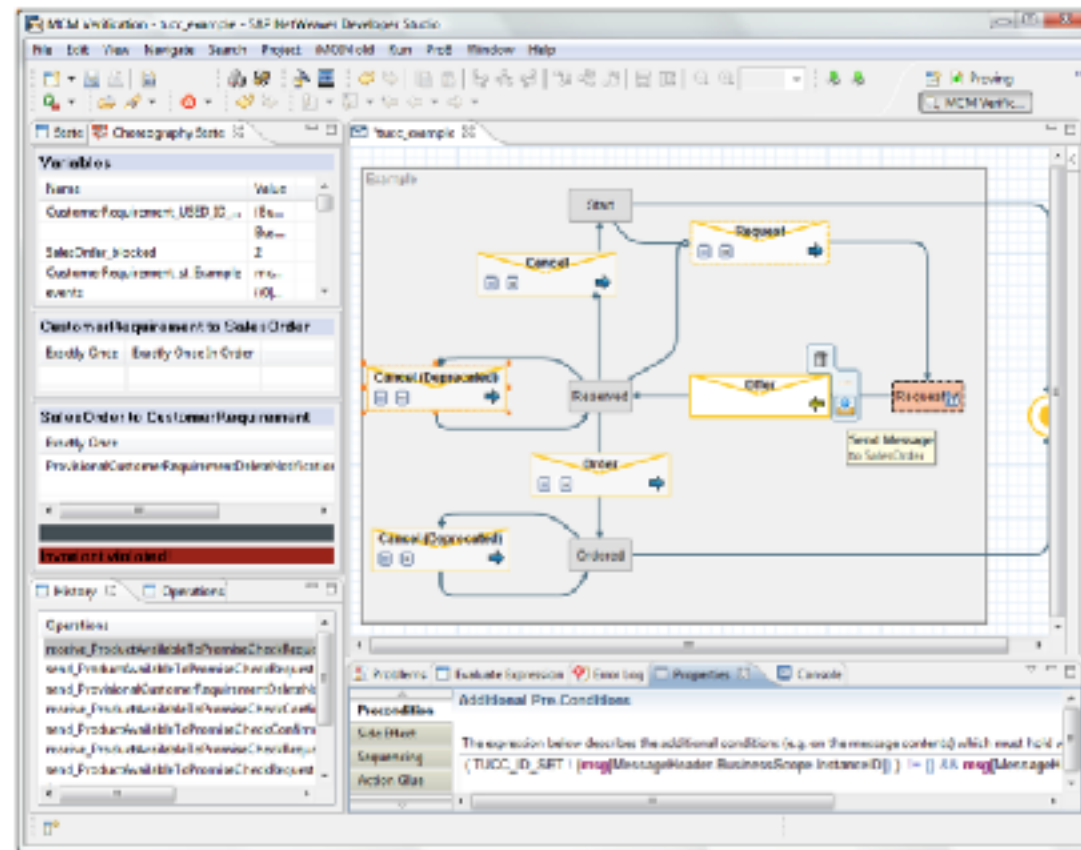
# ProB Uses for Data Validation

- RDV (Siemens; L1, Barcelona,...): used as primary toolchain
- Ovado, Ovado2 (Systemrel, RATP): used as secondary toolchain
- DTVT (Clearsy, Alstom),
- Dave (ClearSy, G&E),
- Olaf (ClearSy, SNCF), ... : used as primary tool chain
- Caval (ClearSy), Rubin (Thales): currently used as only tool chain

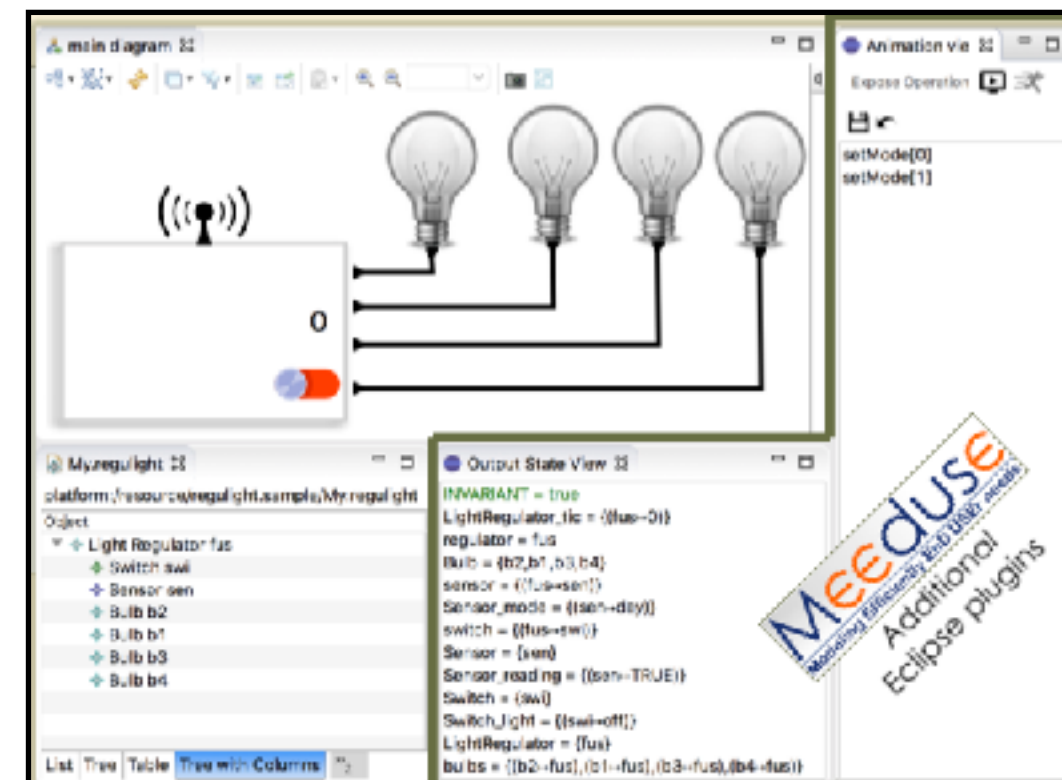


# A lot of (other) tools build upon ProB

## Model checking, simulation, test-case generation, ...



Function	Resource							function comments
	Server_1	Server_2	Server_3	Server_4	Server_5	Server_6	Server_7	
function_A	█							stop function, restart function
function_B		█						start function
function_C		█	█					stop function, restart function
function_D		█						stop function, restart function
function_E			█	█				stop function, restart function
function_F					█			start function
function_G	█							stop function, restart function
function_H	█							stop function, restart function
function_I	█					█	█	stop function, restart function



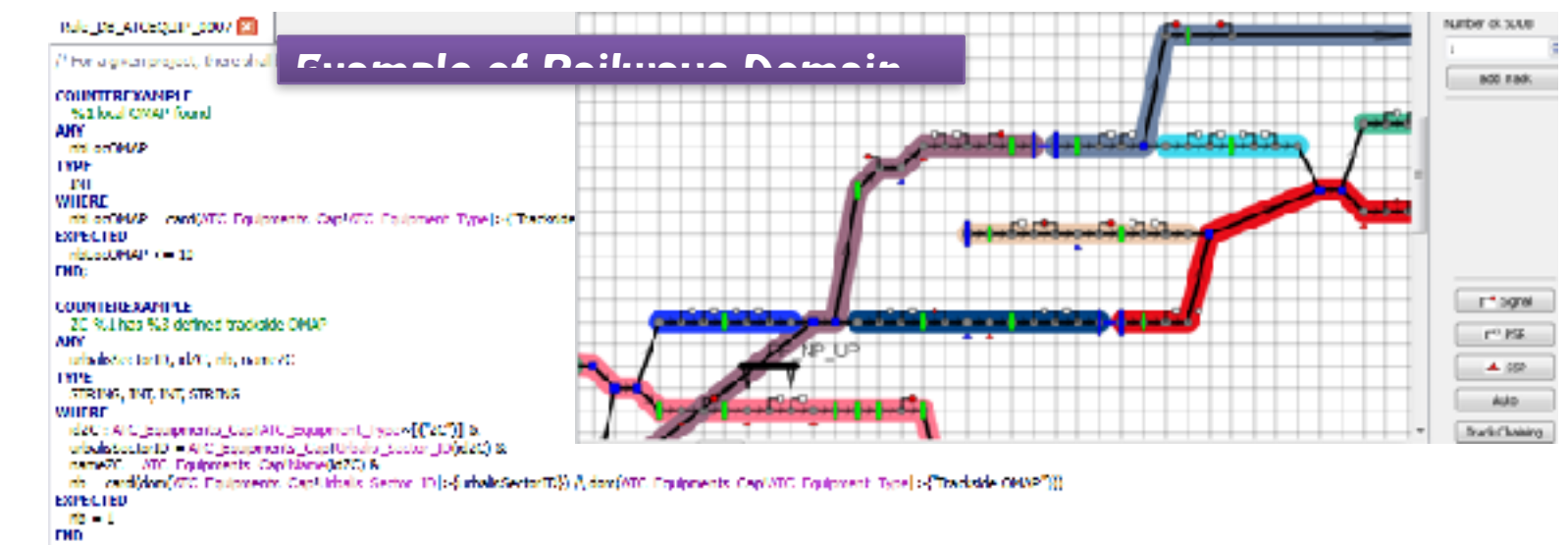
### RUBIN Checker

Choose a project and upload an XML data file:

CH 9.4 (Ed: 14), Rubin 1.0.0-SNAPSHOT

Choose File | STR\_exp01.xml

check



ClearSy Data Solver (Caval), ...



# Thanks to Prolog and Logic Programming

- Convenient to express semantics of specification languages
- Possible to write domain specific solvers, via Prolog's flexible computation rule
- Compact encoding of analysis, optimisation, type checking and verification rules
- Very fast and robust Prolog systems like SICStus Prolog, with efficient co-routines and fast CLP(FD) library

# Conclusion: Prolog

has enabled a tool that



- brings formal mathematics to life
- helps find bugs in safety critical systems
- helps users visualise and understand their models
- is used in industrial applications
- is used for teaching
- is used in research (> 1000 citations)
- is the foundation for many other tools



hhu.

## STUPS Team & Friends

# Thanks for the Support

**Alstom** (F. Mejia,...)

**ClearSy** (T Lecomte, R. Lapostelle, E. Mottin,...)

Siemens

Systerel

**Thales** (N. Nayeri, G. Hemzal,...)

DFG (Gepavas I+II, IVOIRE)

EU (Rodin, Deploy, Advance)

**SICStus Prolog** (Mats Carlsson, Per Mildner)

Jens Bendisposto

Carl Friedrich Bolz

Michael Butler

Joy Clark

Ivo Dobrikov

Jannik Dunkelau

Nadine Elbeshausen

Fabian Fritz

Marc Fontaine

Marc Frappier

David Geleßus

Stefan Hallerstede

Dominik Hansen

Christoph Heinzen

Yumiko Jansing

Michael Jastram

Philipp Körner

Sebastian Krings

Lukas Ladenberger

Li Luo

Thierry Massart

Daniel Plagge

Antonia Pütz

Mireille Samia

Joshua Schmidt

David Schneider

Sherin Schneider

Corinna Spermann

Sebastian Stock

Yumiko Takahashi

Edd Turner

Miles Vella

Fabian Vu

Michelle Werth

Dennis Winter