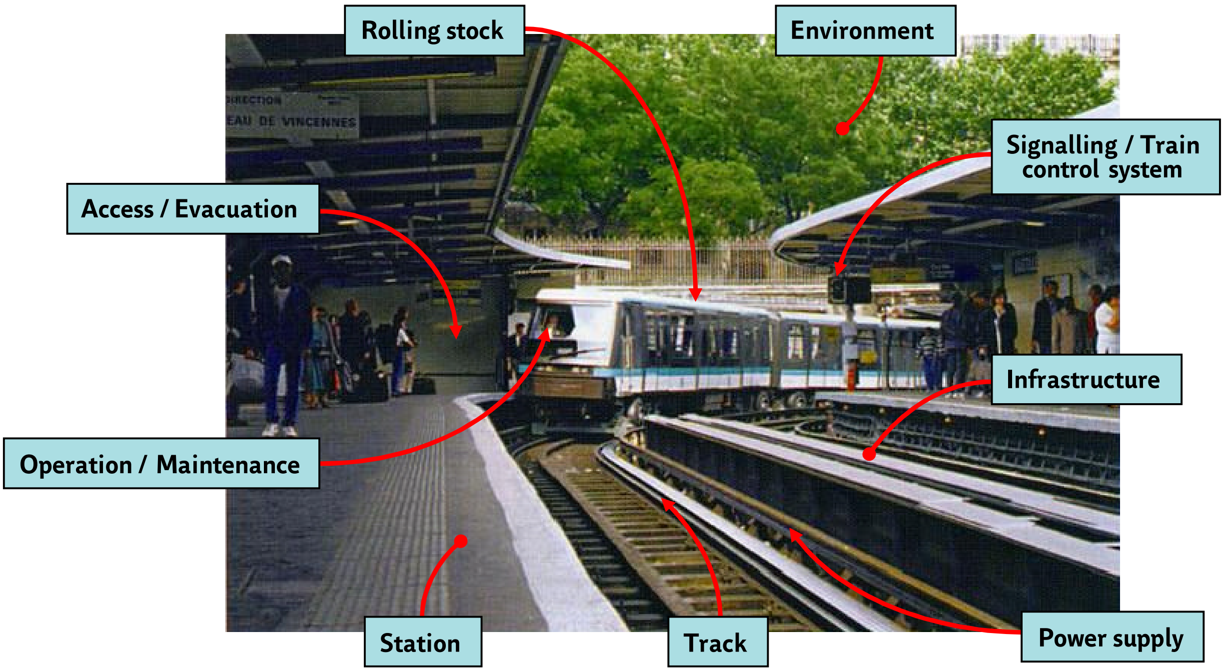
RATP safety approach for railway signalling systems

ReSIST summer School 2007
Pierre CHARTIER

Summary

1. Introduction
2. Hardware fault detection
3. Software fault avoidance

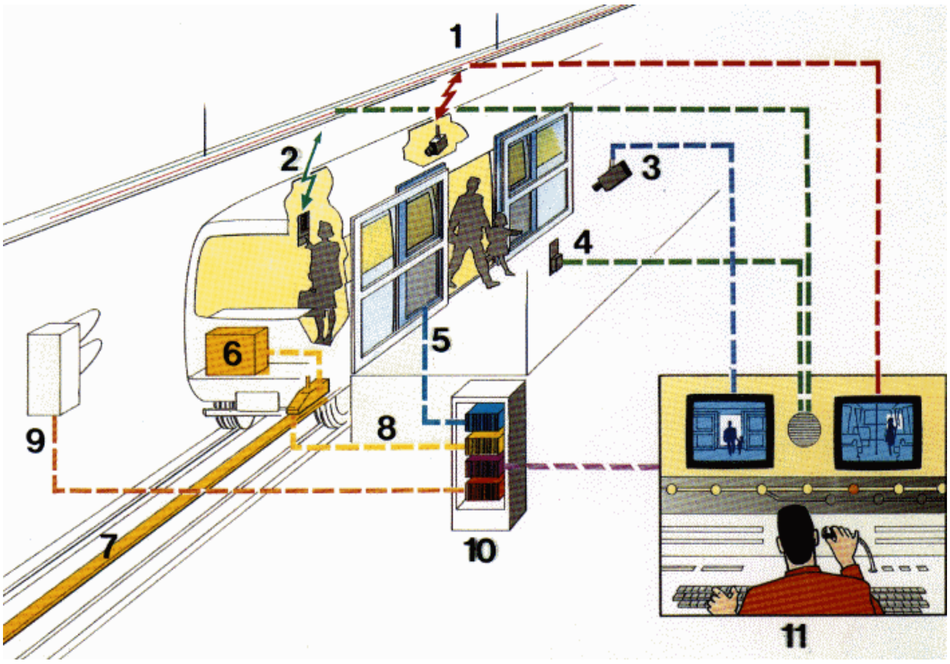
Global railway system



Events to be feared at global system level

- Fire / explosion
- Derailment / overturning
- Panic
- Electrocution / burn
- Collision
- Individual accidents (fall, …)
- Others (terrorist attack, natural disaster, structure breaking, …)

Transport system overview (METEOR example)



1. video in train
2. Inter-phone in train
3. video in platform
4. Inter-phone in platform
5. Platform screen doors
6. onboard equipment
7. transmission
8. track-vehicle communication
9. interlocking
10. trackside equipment
11. operation control center

Railway signalling system

Main protection against collision and derailment

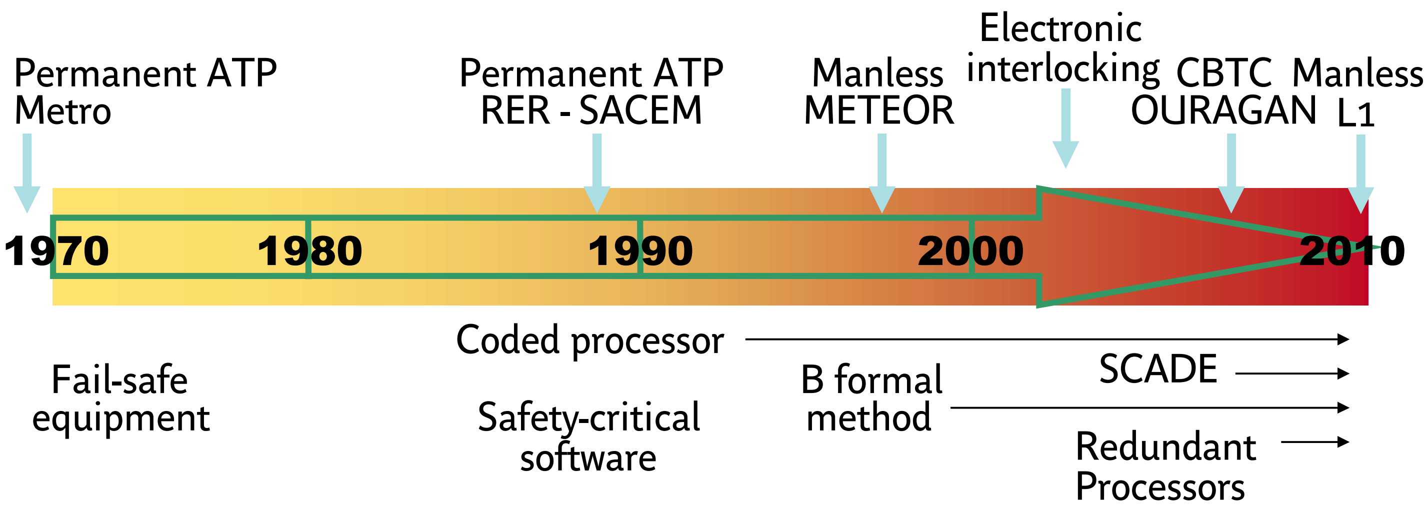
- Safety critical mission

Historically 2 types of system

- Interlocking
- Automatic train protection

Main safety measure : stop all trains and power off the traction power supply

RATP technical evolution



SACEM – RER A

Saturation of RER line A (80's) → SACEM project

- Objective :
  To increase transport offer by raising train frequency

- But incompatibility between
  – train spacing reduction,
  – and traditional signaling

SACEM – RER A

Automatic Train Protection

- Control train spacing
- Control train speed
- Protect switching zones
- Switch between cab signal and trackside signalling

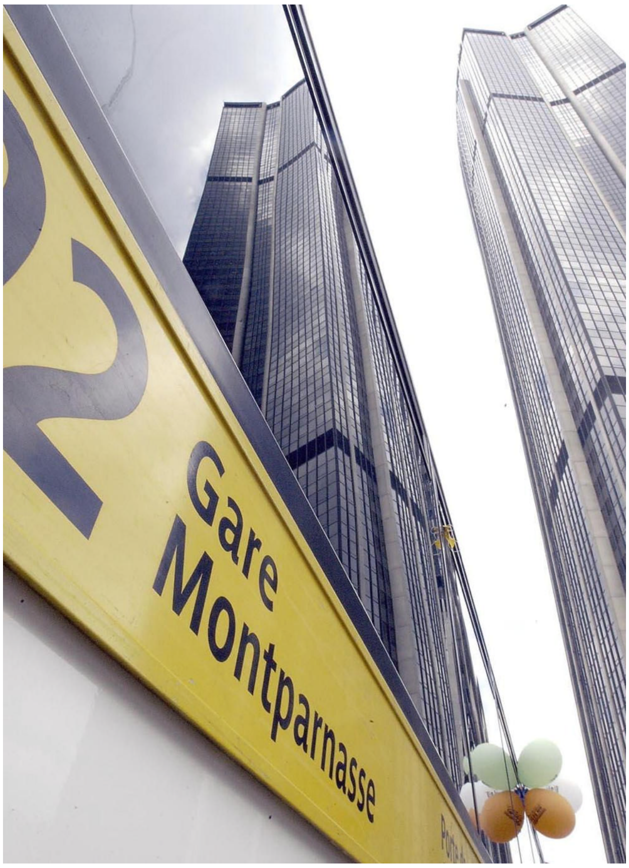
→ First safety-critical computing system in railways

SACEM – RER A

SACEM functions require the use of computers

Two main concerns :

- Detection of errors due to hardware
  → coded processor
- Avoiding faults in software
  → formal methods

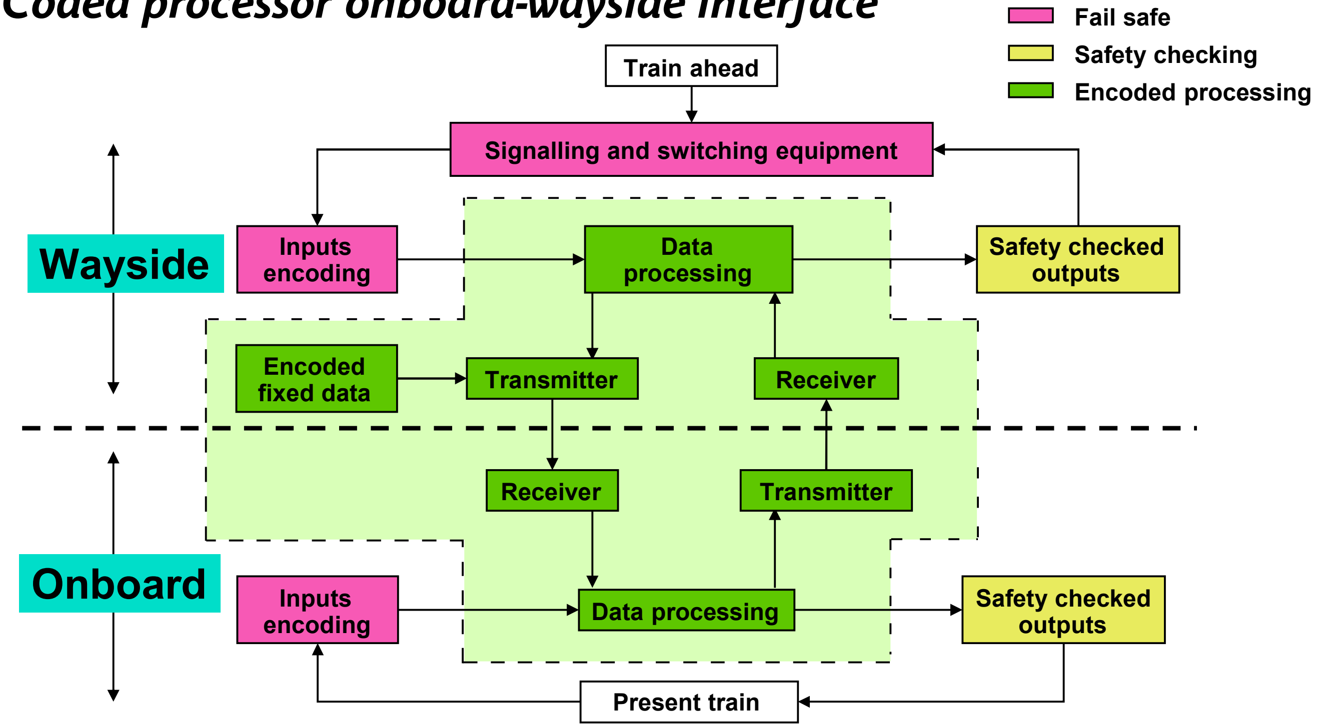


Hardware fault detection

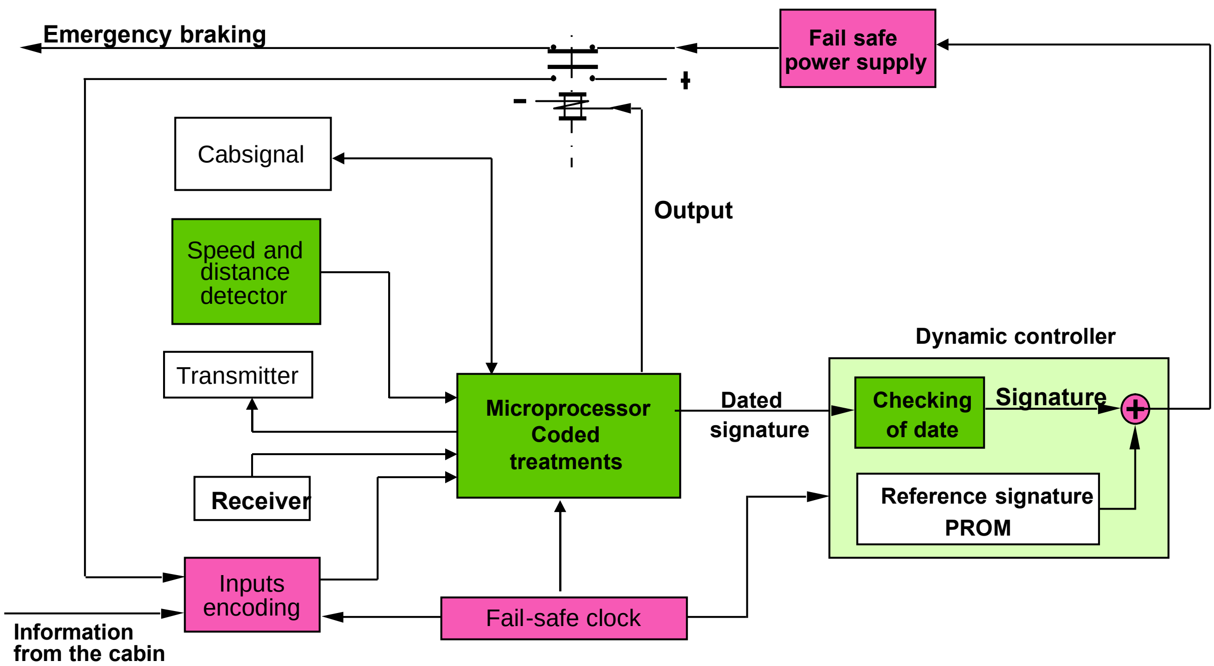
Coded processor – main concepts

- Based on data and program encoding
- Encoding done automatically by specific tools
- Detect run-time errors
- If an error is detected, the hardware sets the system in a fail-safe state

Coded processor onboard-wayside interface



Coded processor



Coded processor – safety data encoding

Data X = functional part X.F and coded part X.C

X.F : N_F bits and X.C : N_C bits

Tasks for the computer

- Acquisition and coding of the fail-safe inputs
- Processing the coded data
- Conversion of coded data into fail-safe outputs
- Setting the system into restrictive state in case of failure

Coded processor – detected errors (1)

Differents kinds of errors :

- Arithmetical error
- Operator error
- Operand error
- Memory « non-refreshed » error
- Branch error

Hardware fault detection

2

Coded processor – detected errors (2)

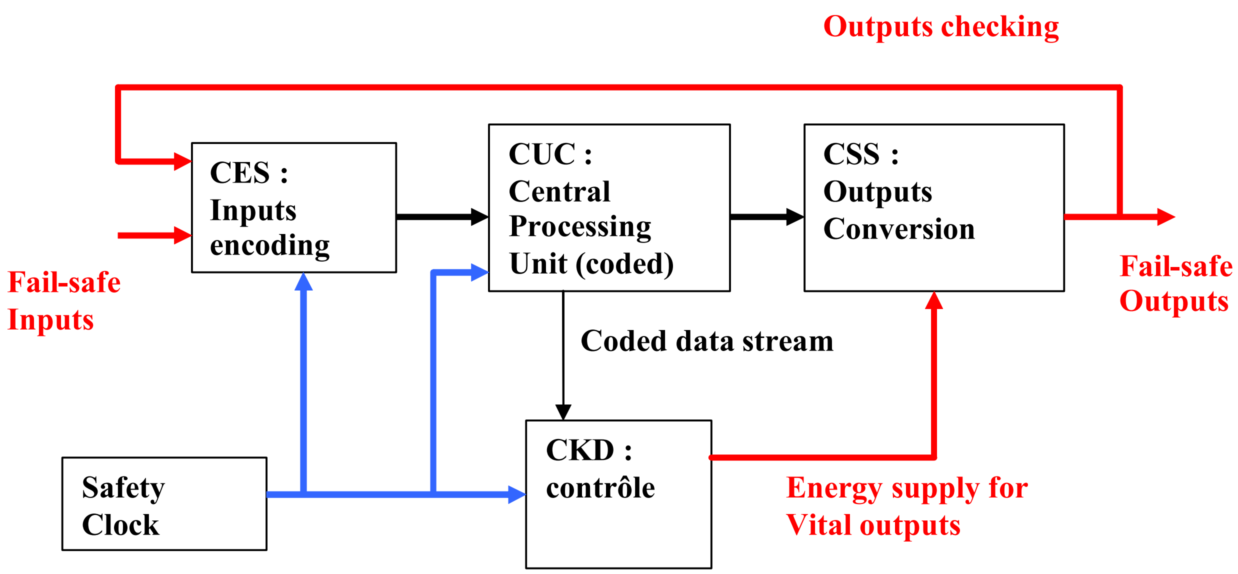
Components of the coded part

- Arithmetical error ==> remainder r_{kx}
- Operator error ==> signature B_x
- Operand error ==> signature B_x
- memory « non refreshed » error ==> date D
- Branch error ==> compensation, tracer

Hardware fault detection

2

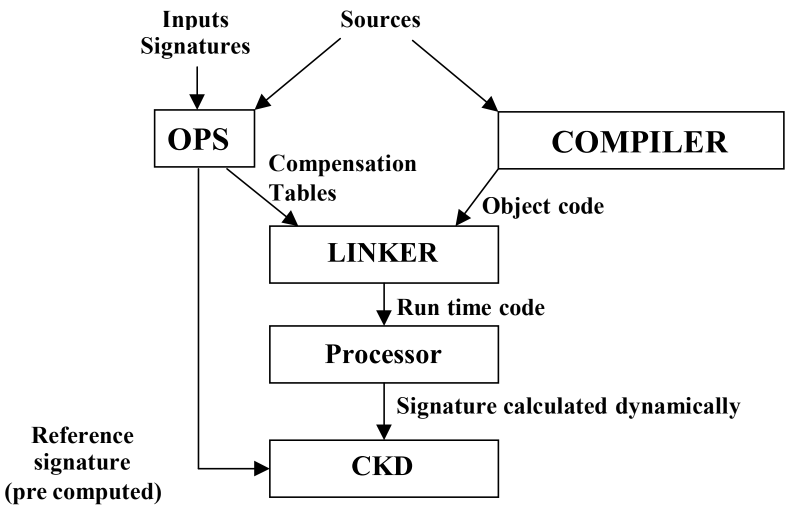
Coded processor – architecture



Hardware fault detection

2

Coded processor – Signature predetermination tool (OPS)



The OPS protects from the compiler failures,

therefore the compiler does not require specific qualification

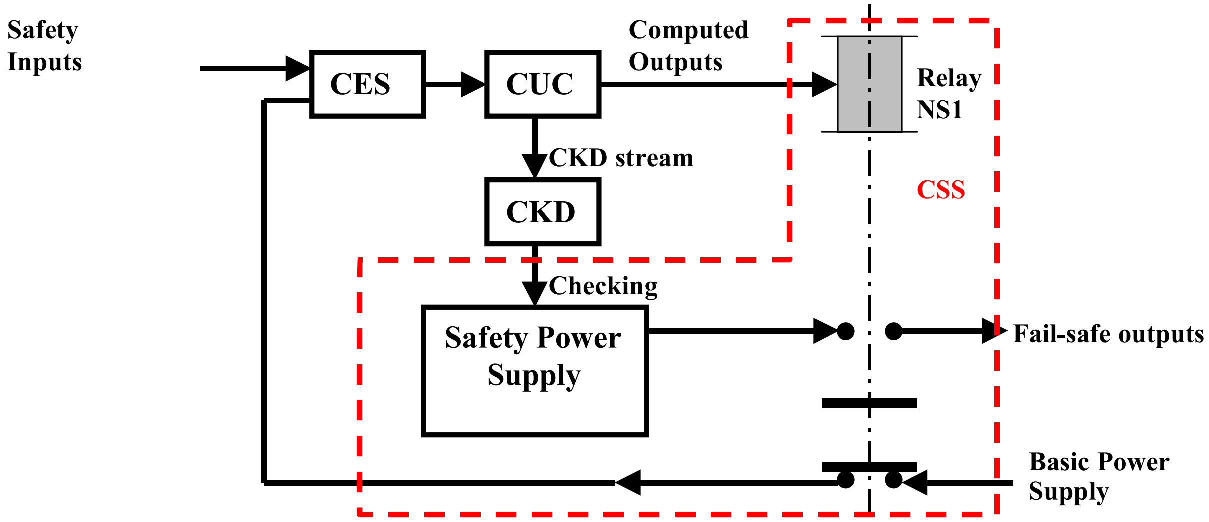
Hardware fault detection

2

Coded processor – safety outputs

Setting of safety outputs

- rereading of outputs



Hardware fault detection

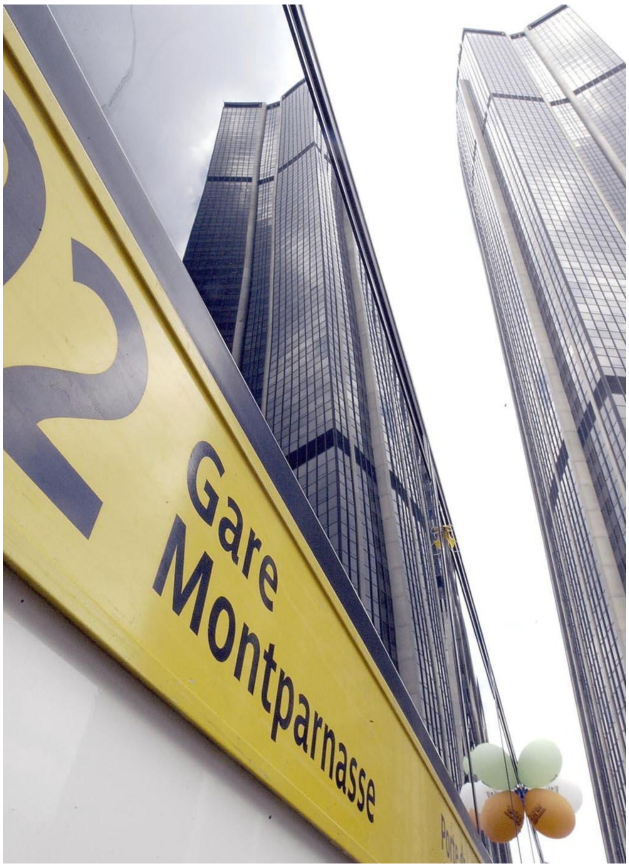
2

Coded processor – safety integrity level

Theoretical result:

- Coded processor alone → 10^{-12} h^{-1}
- Including transmission between onboard and trackside equipment → 10^{-9} h^{-1}

3

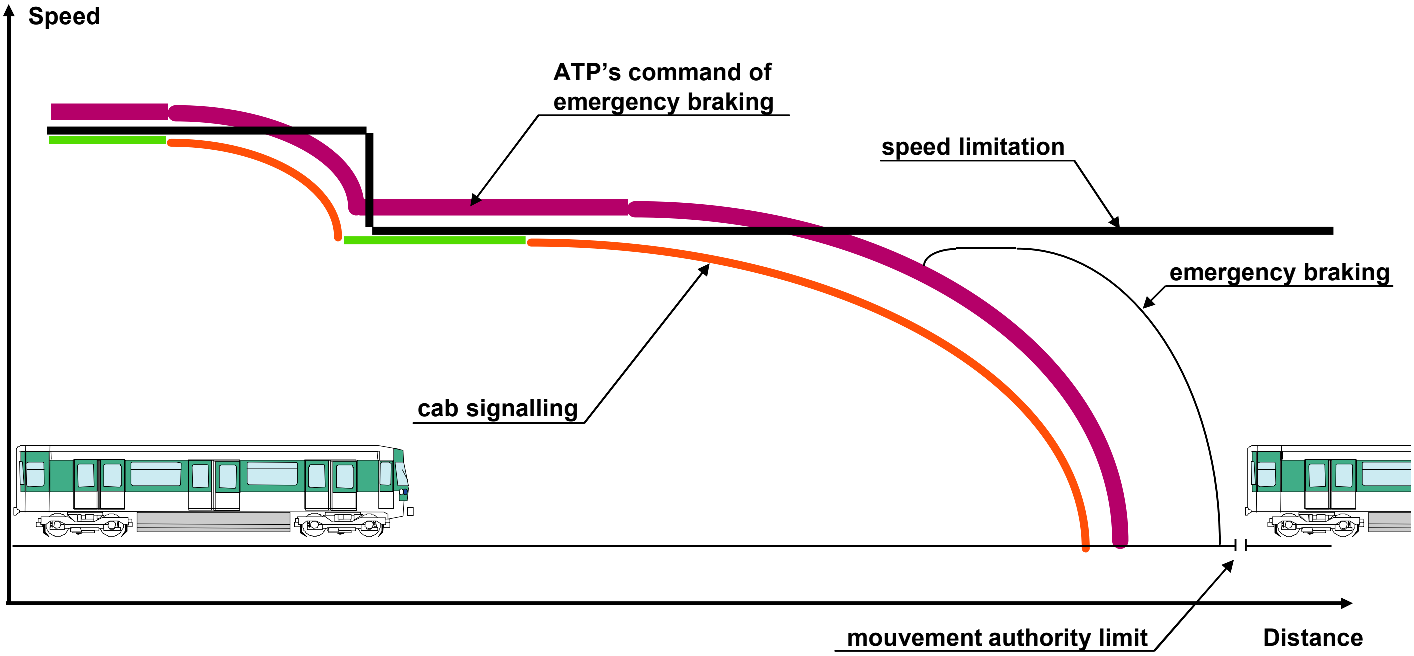


Software fault avoidance

Software fault avoidance

3

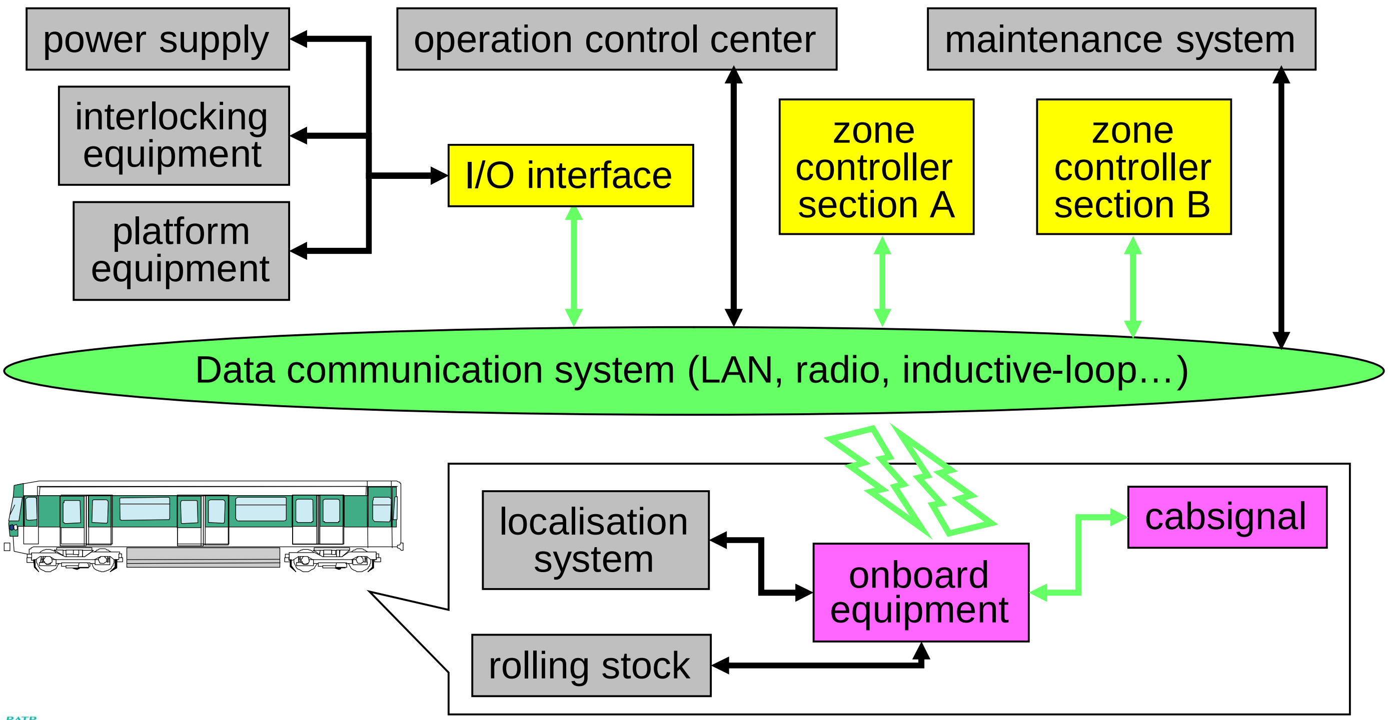
ATP role



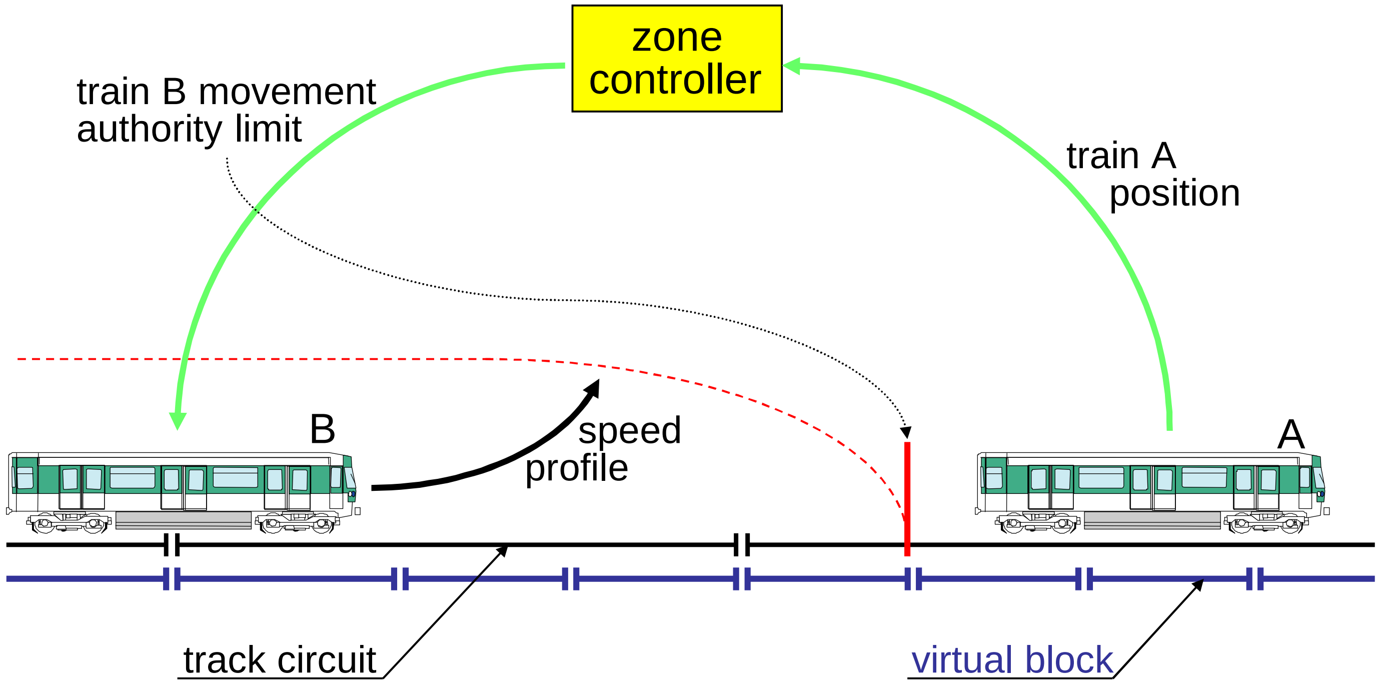
Software fault avoidance

3

Comunication-based train control (CBTC) systems



CBTC operation



Comunication-based train control (CBTC) systems

Automatic Train Protection (ATP)

Automatic Train Operation (ATO)

Automatic Train Supervision (ATS)

Formal methods

1988 SACEM - First safety software in railways

- Usual (unformal) software specification issues
  - lack of global approach with the system designer point of view
  - ambiguous, not legible, not coherent, not complete
- Validation issues
  - no certitude that the functionnal tests are sufficient

1998 First run of the subway line 14 Météor

The B method is used to obtain :

- a reliable and exact software design from specifications to runtime code

B formal method

Goal

- To get a software which meets completely its functionnal specification by construction

Application fields

- Sequential code with no interruptions (real time aspects, low level softwares, operating kernels are not taken into account)

Large spectrum language

- Unified framework and continuous process from specification to code

B formal method

High level language

- Abstract operators for specification needs
- Concrete instructions similar to ADA or C one's

Model oriented approach

- Software = data + properties + operations

Refinement process

- Translation of the abstract machines into concrete modules, and finally into code

Proof obligations

- Conditions to check to ensure a strict mathematical construction of softwares

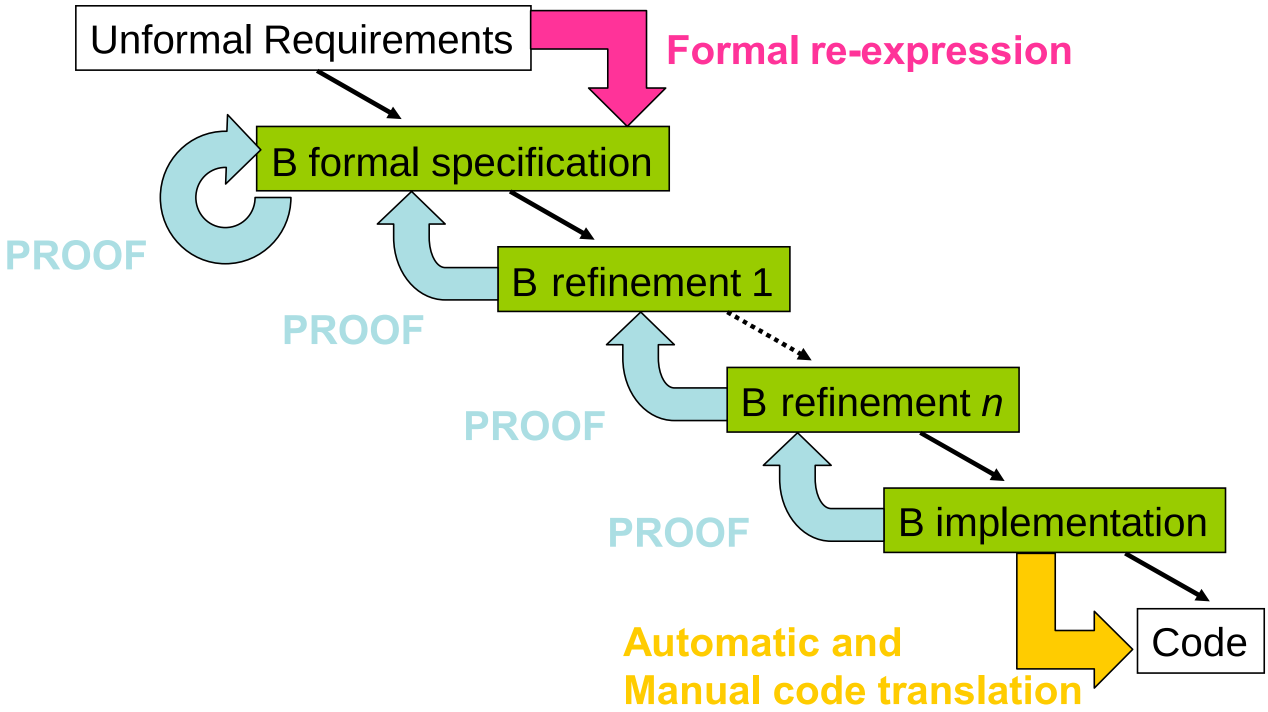
B formal method – examples of safety properties

- Only equipped train which is located and in automatic mode can have a target.
- The trains locations computed by the SWE must be correct with the actual trains locations on the line.

Software fault avoidance

3

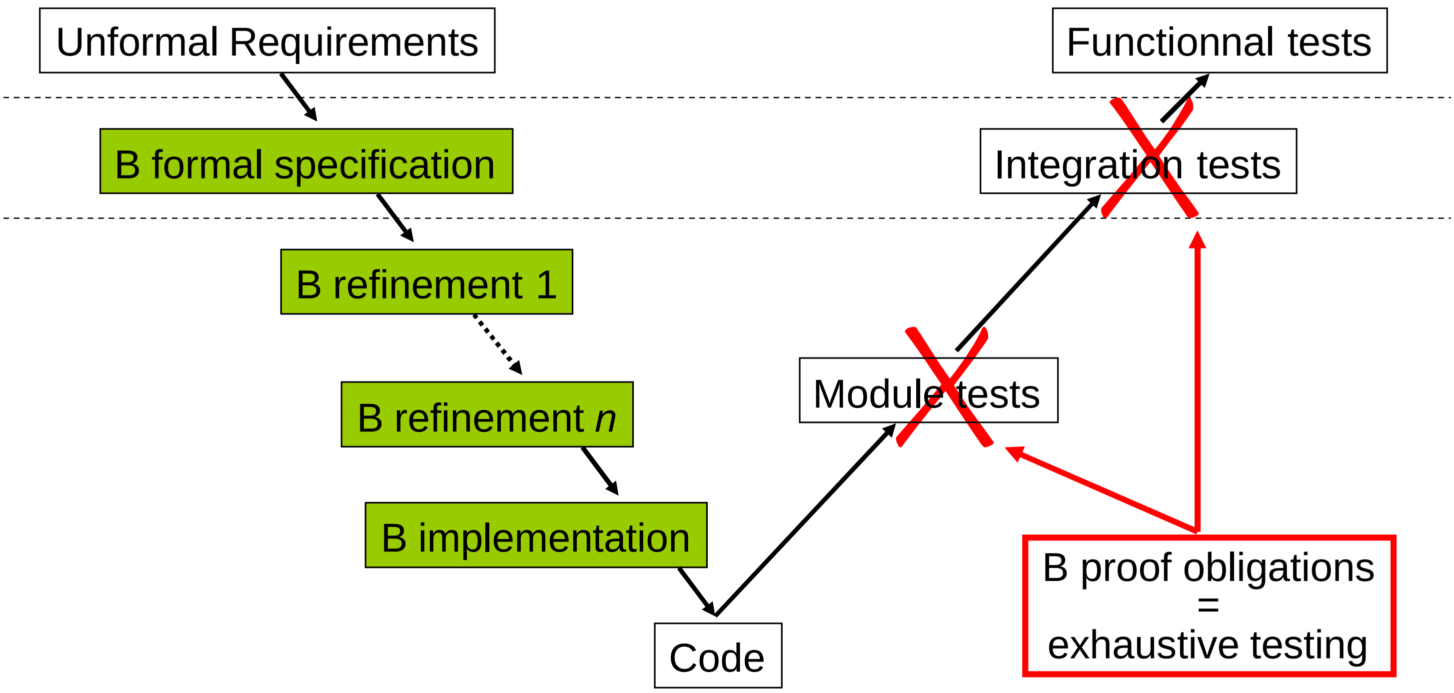
B development process



Software fault avoidance

3

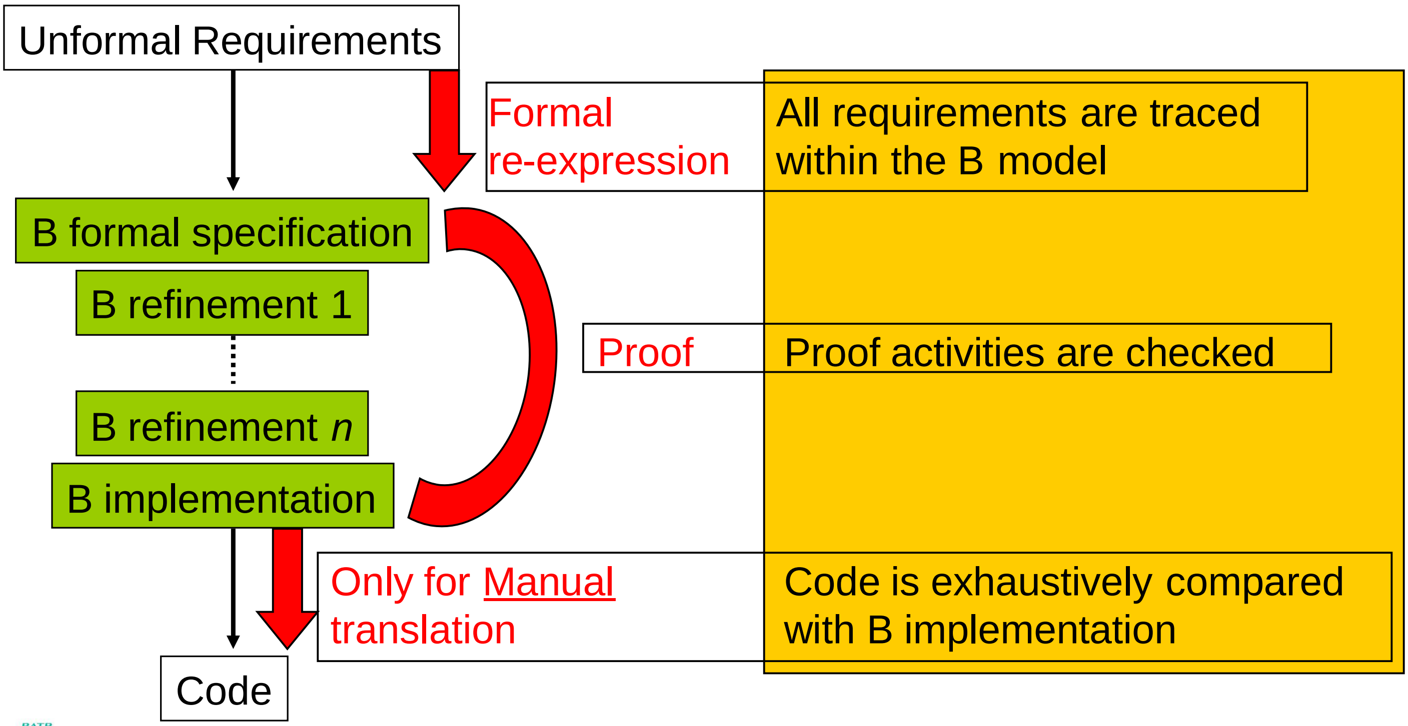
B verification process



Software fault avoidance

3

B validation process



Software fault avoidance

3

B industrialisation

AtelierB© : An industrial tool to specify, refine, implement and prove B models

Statistics about Météor B model

- 1150 B components
- 115 000 lines of B code
- 27 800 Proof Obligations (all proved)
- 86 000 lines of « safe » ADA code

Software fault avoidance

3

B today in railway industry

Used by two railway leaders : SIEMENS and ALSTOM

Recent projects :

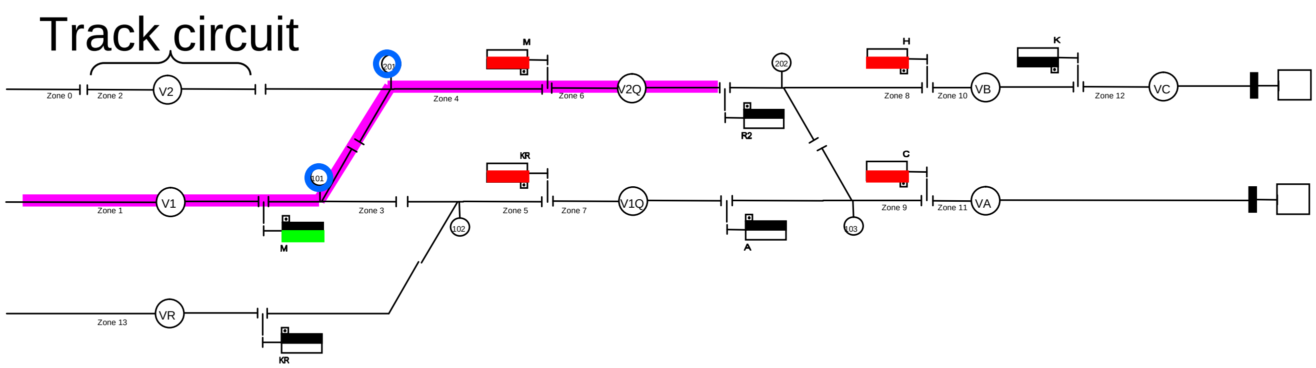
- Canarsie Line (New-York),
- North East Line (Singapour)

Projects size has increased more than twofold

Software fault avoidance

3

Interlocking system



Route V1 to V2Q

Signals M (V2), H, KR, C

Switches 101, 201

Signal M (V1)

Software fault avoidance

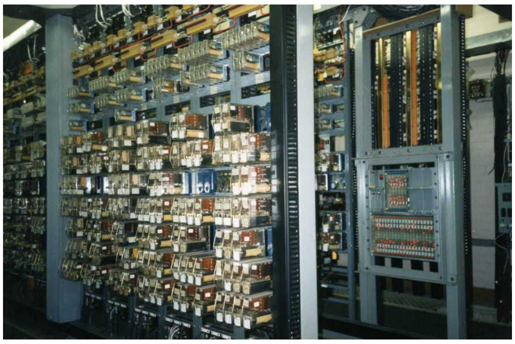
3

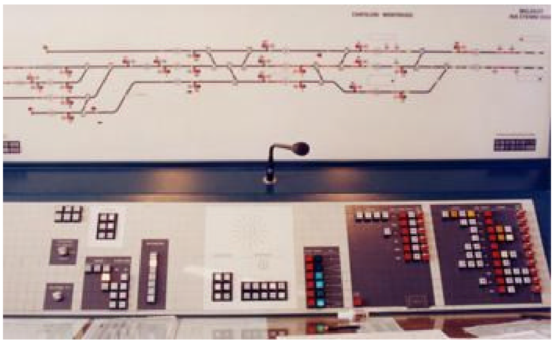
Relay interlocking

Main technology on RATP network

- Fail-safe relays
- Man-machine interface with button/switch control panel

Increasing cost and expensive reconfiguration

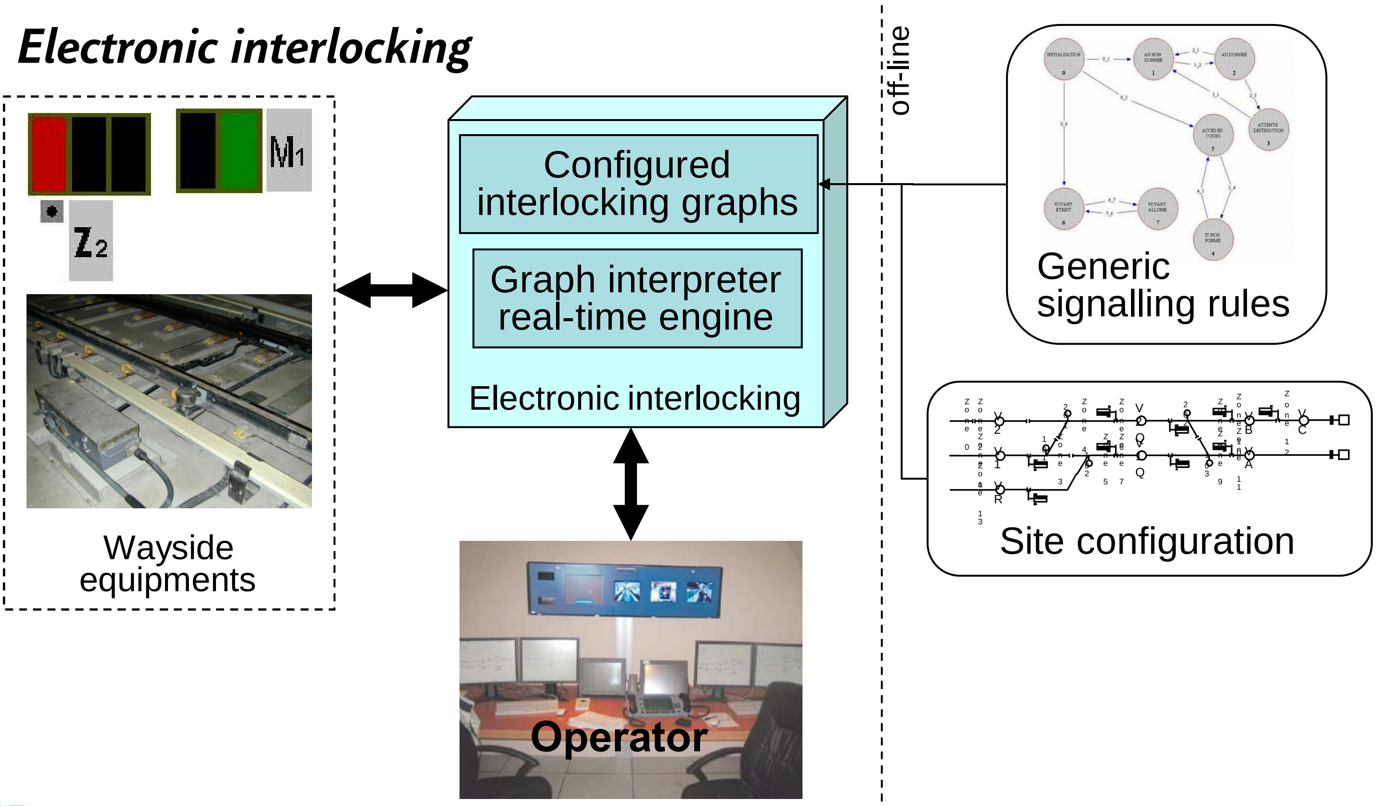




Software fault avoidance

3

Electronic interlocking



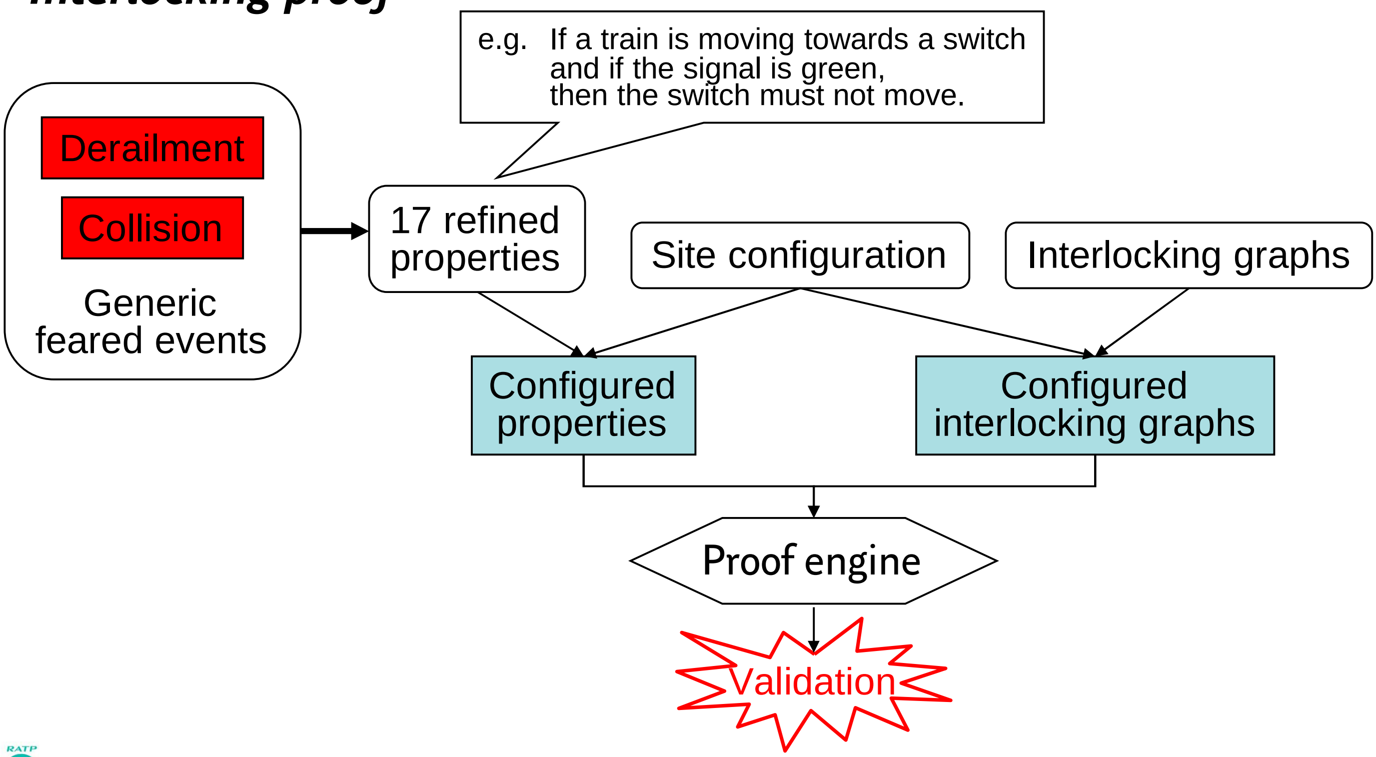
Interlocking validation

Issue: how to be convinced that any combination of generic graphs for any site configuration is safe ?

- Heavy testing for both supplier and RATP on site configuration

To reduce test effort for next interlocking sites, formal proof of safety properties has been considered.

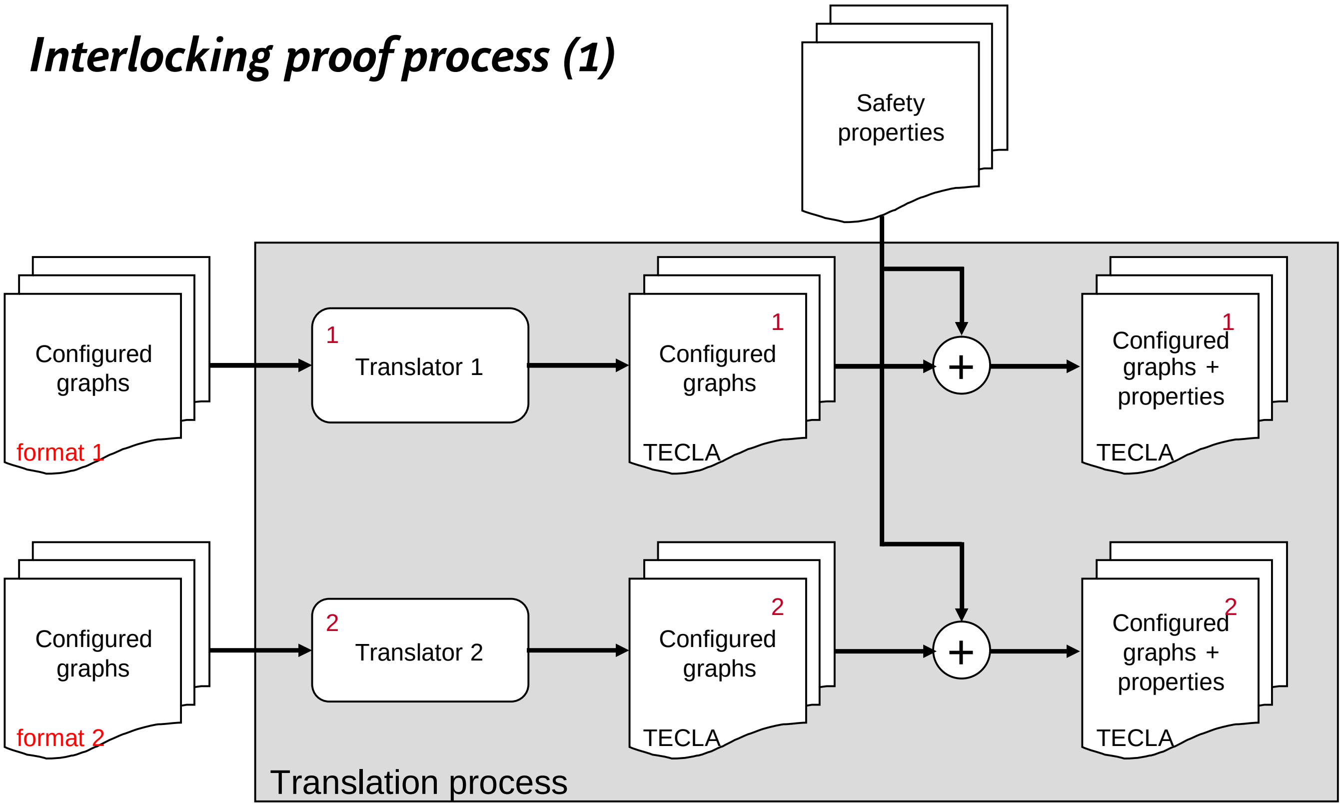
Interlocking proof



Software fault avoidance

3

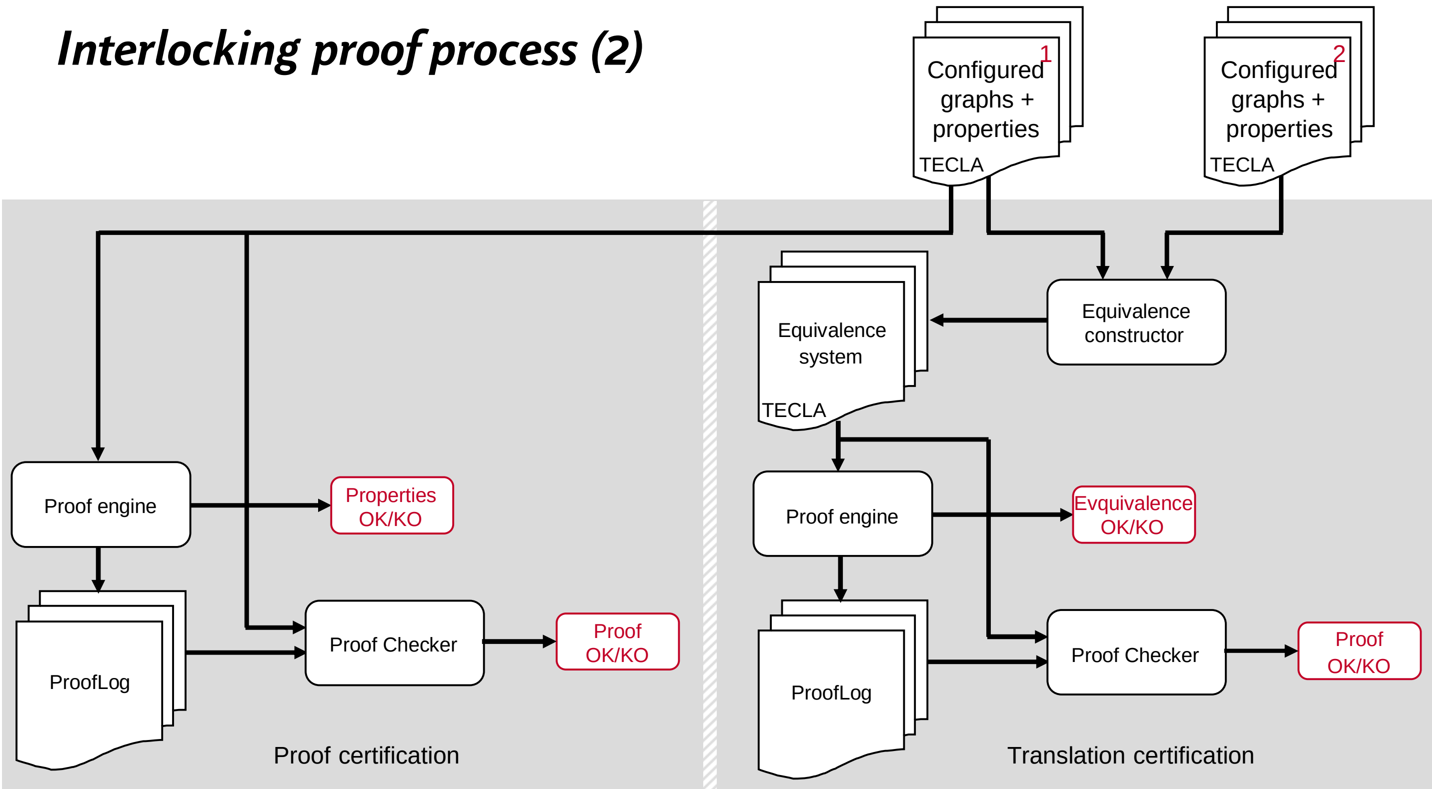
Interlocking proof process (1)



Software fault avoidance

3

Interlocking proof process (2)



Interlocking proof

The proof engine (from Prover Technology) is based on combination of SAT techniques and other automatic proof techniques.

Work in progress

- Feasibility is established
- Complete proof of a real interlocking configuration is expected in a few months

Apparition of SCADE tools in railway industry

For a few years, SCADE has found favour with railway industry

- fitted for designing command-control systems
- reduces developement cost
- facilitates communication between specialist engineers and software engineers

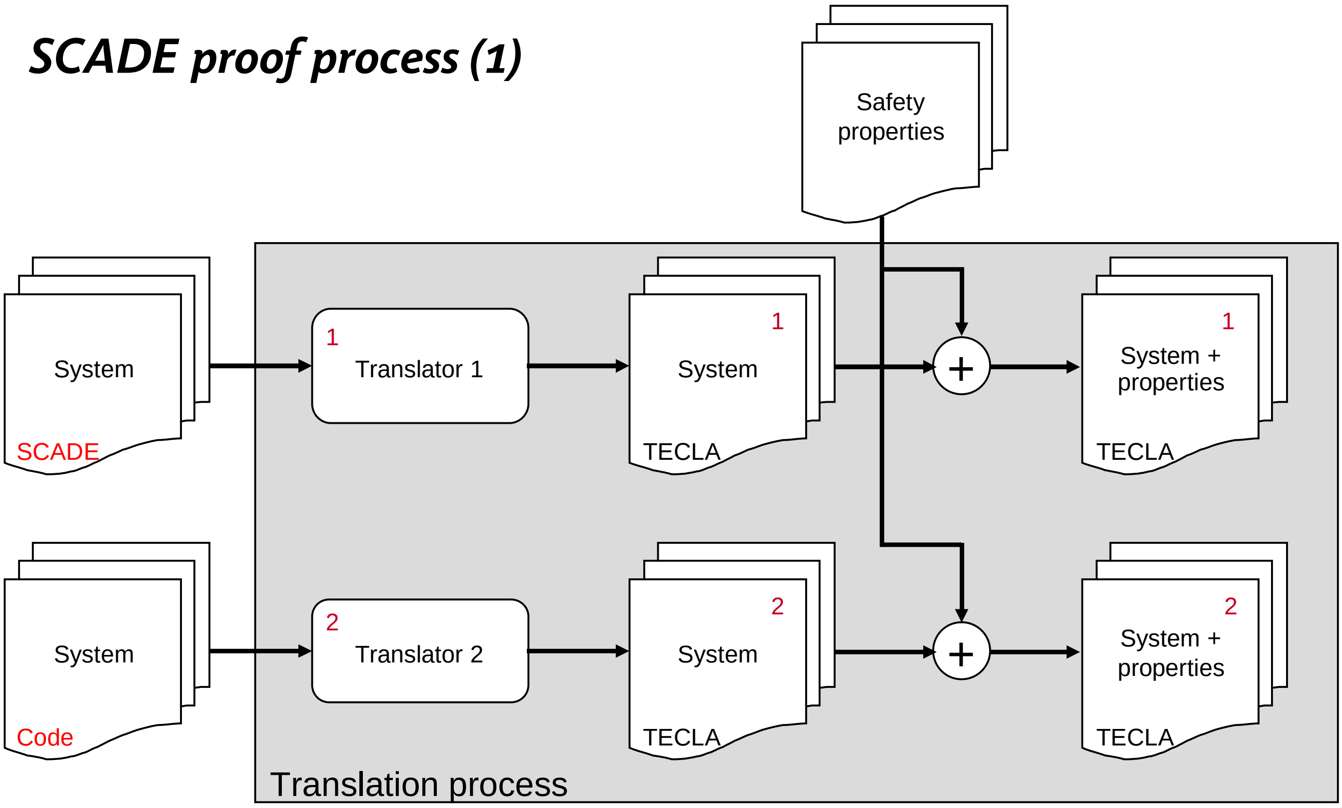
SCADE brief overview

- based on a declarative synchronous language Lustre, encapsulated in graphical representation
- Software = variables + equations
- Time is discrete $(var_n)_{\mathbf{N}}$
  clocks, temportal operators (pre, when, …)
- Equations between inputs and outputs
  $out_n = \Phi(in_n, \ldots, in_{n-p}, var_n, \ldots, var_{n-q})$

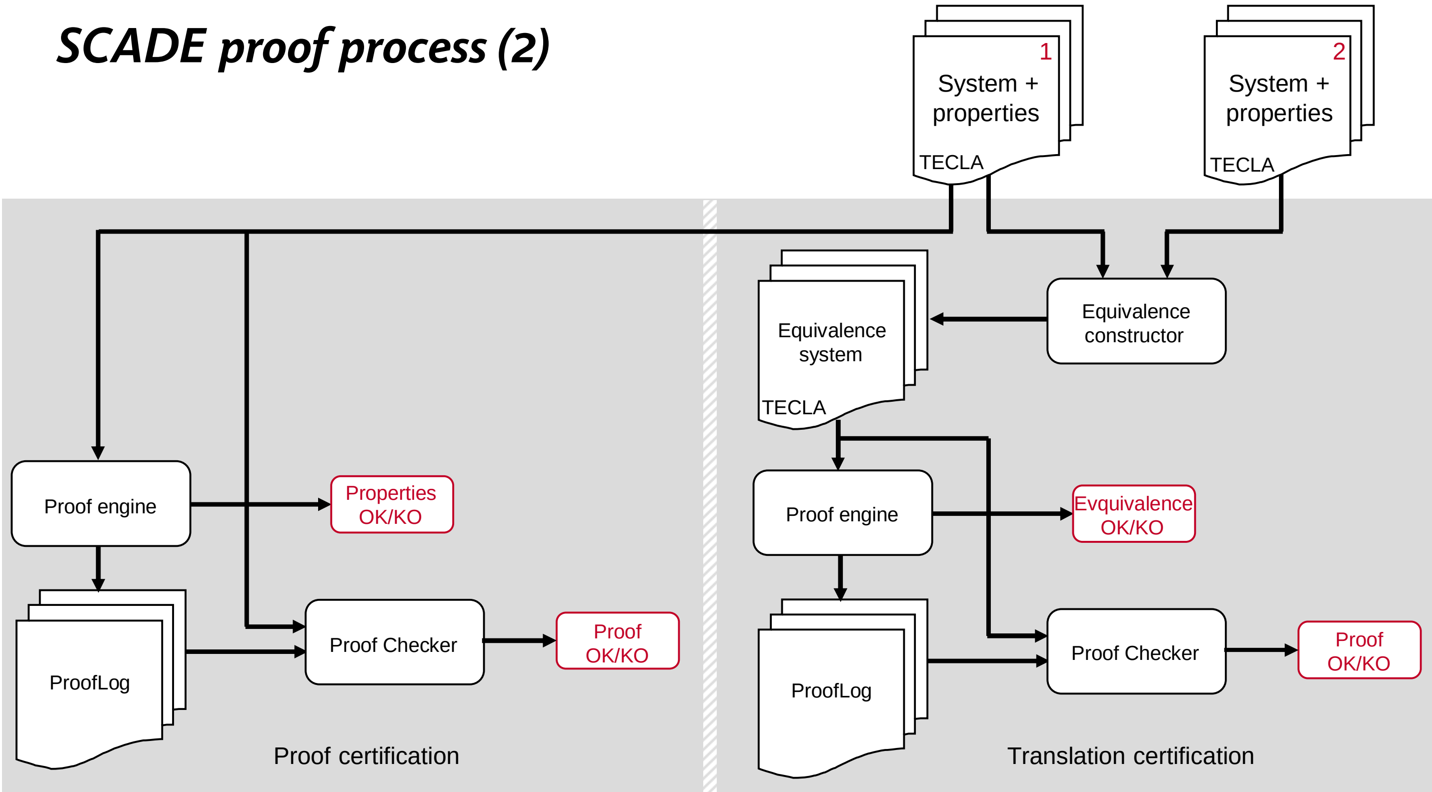
Software fault avoidance

3

SCADE proof process (1)



SCADE proof process (2)



SCADE proof

Example of safety property :

- Two distinct trains must not cross their movement autority limit

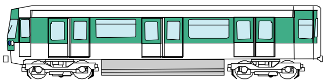
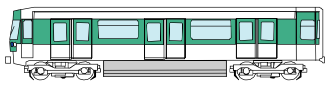
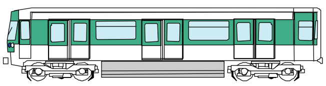
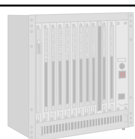
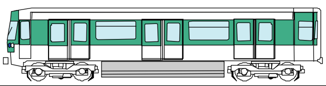
Work in progress

- Feasibility on a real site configuration
- System requirements specification coverage
- Method to complete proof when safety properties are not totally proved

Formal methods …

- reduce drastically test effort
- provide a high level of quality and safety for software
- are applicable to industrial software projects
- but have to take more into account the practical aspects for using them (cost, competence, …)

RATP renewal program: software development methods

	OURAGAN CBTC	Manless CBTC
B method Coded processor	 L3  L5 WaySide Equipement	  L1
SCADE Redundant processors	 L5  L3   L13	