ADVANCED DESIGN AND VERIFICATION ENVIRONMENT
FOR CYBER-PHYSICAL SYSTEM ENGINEERING
www.advance-ict.eu

D1.5 - FINAL REPORT ON APPLICATION IN RAILWAY DOMAIN

ADVANCE

Grant Agreement: *287563*
Date: *30/11/2014*
Pages: *38*
Status: *Final*

Authors: *Fernando Mejia, Alstom; Minh-Thang Khuu, Systerel; Asieh Salehi,U Southampton*

Reference: *D1.5*
Issue: *4*

Partners / Clients:

SEVENTH FRAMEWORK PROGRAMME

FP7 Framework Programme

European Union

Consortium Members:

University of Southampton | ***Critical Software Technologies*** | ***Alstom Transport*** | ***Systerel*** | ***Heinrich Heine Universität*** | ***Selex ES***

Contents

List of Tables

List of Figures

1. Introduction

The case study in the railway domain of the ADVANCE project had four main objectives. The first objective was to develop with ADVANCE technology a formal model of an interlocking dynamic controller (IXL-DC) that enforces interlocking's system-level safety requirements. The second objective was to demonstrate the feasibility of a formal solution independent of the implementation technology of the controlled interlocking and of the complexity of the interlocked area. The third objective was to integrate ADVANCE technology into Alstom's system development process to take full advantage of formal development while being compliant with the international railway certification standards (CENELEC). And the fourth objective was to provide feedback to ADVANCE technology developers and contribute to help bring ADVANCE technology to an industrial maturity level.

This document is the final report of the case study in railway domain. It provides a complete overview of the work done to achieve the above objectives and of the obtained results. It details more particularly the work of the last period of the project related to hazard analysis (section 3), model composition/decomposition (section 4.1.2) and automatic model simulation (section 4.2.2). The document is organised as follows.

The second section recalls the problem of exhaustive verification of interlocking systems. It explains the existence and the design of the IXL-DC as a solution to this problem that avoids the disadvantages of model checking based solutions by reusing concepts applied successfully to another safety critical component of a railway signalling system.

The third section presents the hazard analysis done using STPA, the hazard analysis method promoted by ADVANCE. The first part of the section recalls the system control theory and the accident causation model underlying the method. The second part presents extracts of the hazard analysis that illustrate the different steps of the method, notably, the control structure on which the analysis is based and two scenarios of unsafe control.

The fourth section presents the development of the formal model of the IXL-DC. It involves four parts. The first part presents the refinement and decomposition steps that permitted the creation of the model of the IXL-DC with the desired architecture while controlling the number and the complexity of the generated proof obligations. The second part of Section 4 presents the validation of the IXL- model using manual and automatic animation. In particular it explains how factory tests of actual signalling systems were used to validate dynamic features of the model that cannot be checked by other means. The third part of Section 4 deals with formal verification (proof) of the model of the IXL-DC and explains how "theories" were useful to ease that proof. The fourth part of Section 4 explains how Event-B system models can be translated into Classical-B software models when moving from system development to software development. This transition enables formal development of systems from system specification to software implementation.

The last section of the deliverable, the conclusion, assesses the results of the case study as regarding its objectives and the contribution of ADVANCE as regarding system development. The section terminates with the presentation of the exploitation the results of the case study.

ADVANCE Deliverable D1.4 [4] presents the system development process we have defined by integrating ADVANCE technology in Alstom's system development process and by taking inspiration from Alstom's software development process which already integrates the Classical-B method. Deliverable D1.4 explains also the contribution of ADVANCE technology to the certification process based on CENELEC standards.

2. Interlocking dynamic controller

A railway signalling system for an urban network involves three main subsystems.

An automatic train supervision system (ATS) that regulates the traffic of trains; typically, the ATS subsystem controls the routes and the timetable of the trains.

An automatic train control system (ATC) that operates and protects the trains. An ATC involves an automatic train operation subsystem (ATO) that controls trains so that they respect their timetable and destination, and an automatic train protection subsystem (ATP) that prevents accidents, for instance, rear collisions and accidents due to inappropriate speed or inappropriate management of train doors.

An ATC also involves an interlocking system (IXL) that sets and locks routes for trains in order prevent, for instance, head-to-head and side collisions.

An IXL is a safety critical subsystem that must enforce safety requirements. But the use of conventional techniques (tests) to ensure that it does this effectively is not totally satisfactory because conventional techniques cannot cover, except in very rare occasions, all the possible situations. That is why other approaches were developed to deal with that problem. Notably the one developed by RATP (Paris transport operator), Thalès (IXL supplier) and Prover Technology (technology provider) based on, to simplify matters, model checking technology. In this approach all the possible executions of the actual IXL are formally reproduced and analysed in order to check that none of them produces outputs creating a state violating safety requirements on route setting and locking.

This approach has undisputable advantages: it is intrusive neither in the development of IXL nor in its execution and it is applicable by IXL experts, not only by verification technology experts. However, in our judgment, it has two drawbacks. First, it is sensitive to the configuration and the operating rules of the network for which the IXL is designed. This means that the richer the configuration and the operating rules are the more numerous are the IXL executions. And the number increases exponentially. Therefore, the approach is inapplicable to IXL controlling medium-large and large network configurations. Second, the approach relies on the implementation technology of the IXL to which it is applied. This means that the analyser of an IXL must simulate accurately the behaviour of the execution engine of the IXL in order to reproduce and analyse its possible executions. Therefore, the development and validation of the analyser of an IXL whose execution model is not as simple as a sequence of Boolean equations (e.g. concurrent automata, SSI interpreter) is difficult.

RATP successfully applies this technology for the acceptance of IXLs provided by Thalès for the Paris Metro. Moreover, RATP demands other potential IXL suppliers to achieve the same level of verification of their IXL as that obtained by RATP on Thalès' IXLs with its own technology.

To meet this demand, Alstom has invented a solution inspired by the design and the development methods of its ATC systems.

As said above, an ATC involves an ATO that controls the movements of trains and an ATP that protects the movements of trains. In other words, the ATP checks dynamically (at execution time) that the ATO does not issue controls that create a state contrary to safety requirements on train movements. In this architecture the ATO is not considered as safety critical while the ATP is. Furthermore, to be easily adapted to different configurations, the ATP was designed in two parts. A generic part implements the dynamic verifications performed by the ATP and is parameterised by configuration data. A specific part contains the values of configuration data of a particular application. The generic part is developed and validated once and for all. The specific part is instantiated and validated for every particular project.

To ensure that the ATP is safe, the software of its generic part and the properties that it must meet are specified formally in Classical-B language and, using mathematical proof, it is verified

exhaustively that these properties are consistent and fulfilled by the final code. This mathematical proof relies on the properties of configuration data, not on their particular values; this verification is therefore insensible to the specificities of particular applications.

Following the ATO/ATP schema we decided on a new component, i.e. the interlocking dynamic controller, which would play vis-à-vis the interlocking system the role played by the ATP vis-à-vis the ATO. That is to say, the IXL-DC would check dynamically that the controls issued by the IXL do not produced a state contrary to safety requirements on route setting and locking. Like the ATP, the IXL-DC would be made of a generic part and a specific part and, also like the ATP, the IXL-DC would be formally developed; except that we want to start formalisation earlier, at system level, in order to verify that IXL-DC meets system-level safety requirements and avoids route setting and locking related accidents.

We were interested by technology centred on Event-B because we want to promote the use of formal methods for system design and because Event-B and Classical-B, which we use for software development, share a large number of concepts and are complementary. The other reason that led us to undertake this case study is that ADVANCE develops a set of consistent and complementary methods and tools for the creation, verification and validation of Event-B models that might improve significantly the quality of designs and the productivity of engineers.

The following sections describe the work done to create, validate and verify the Event-B model of the IXL-DC; the methods and tools provided by ADVANCE that were used and the way they were used.

3. Hazard analysis

This section presents the hazard analysis activity carried out during the third period of the project.

Completeness and accuracy of safety requirements on a safety critical system are paramount. Indeed, a safety critical system is authorised for operational service only if it is demonstrated that it meets all safety requirements. Accordingly, the effectiveness of hazard analysis methods of identification and definition of requirements is also paramount.

ADVANCE promotes the hazard analysis method called STPA (System-Theoretic Process Analysis), defined and popularised by Nancy G. Leveson and her group at MIT. STPA is based on the accident causation model called STAMP (System-Theoretic Accidents Model and Process).

Both, STAMP and STPA are described and exemplified in great depth in [11]. Most of the following presentations are quotes from that book.

3.1 STAMP

STAMP is an accident causation model based on three basic concepts – safety constraints, a hierarchical safety control structure and process models – and basic system theory concepts.

Safety constraints

STAMP considers that events leading to accidents occur because safety constraints were not successfully enforced. Safety constraints on a system are imposed by the laws of physics, the regulatory and organisational frameworks, the systems with which it interacts, the functions it performs and design and development decisions.

System-level constraints are first identified and responsibility for enforcing them is divided and allocated. Then, during system design and development, system-level safety constraints are broken down and sub-constraints are allocated to the system components.

Hierarchical control structures

STAMP considers systems as hierarchical control structures where each level imposes constraints on the activity of the level beneath it. The standard control structure involves four components organised as shown in Figure 1.

The *controller process* issues control actions implemented by *actuators* that affect the state of the *controlled process*. *Sensors* capture changes in the state of the controlled process and transmit them to the controller process which uses this feedback information to issue new actions to keep the controlled process in the desired safe operational state.

Process models

In STAMP the controller process has a model of the controlled process. The controller process maintains this model with the feedback information provided by the sensors and, based on this model, determines the control actions.

Accidents in STAMP are violations of safety constraints that were not adequately enforced by control actions because the model of the controlled process in the controller process departs from the actual controlled process. This discrepancy is the source of the four possible causes of accidents:

- A control action required for safety was not provided.
- An unsafe control action was provided.
- A control action required for safety was provided too early or too late or in the wrong sequence.
- A control action required for safety was stopped too soon or applied too long.

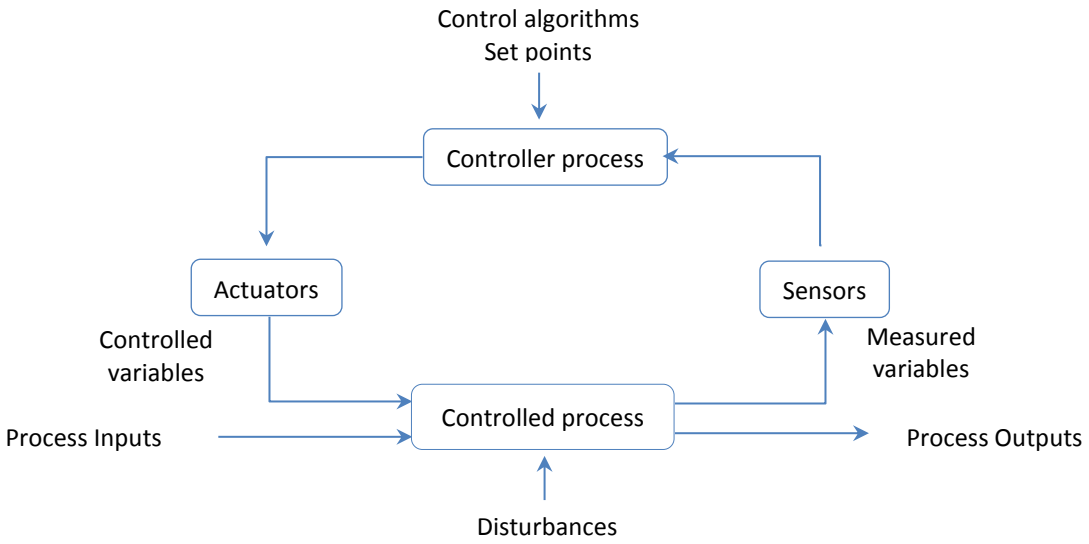

Figure 1: Standard control loop

3.2 STPA

STPA is a hazard analysis technique aiming at accumulating information about how the behavioural safety constraints, derived from the system hazards, can be violated. More precisely, STPA aims at identifying scenarios that can lead to accidents, so they can be eliminated before damage occurs. STPA relies on the STAMP accident causation model described above and on the following standard safety related activities of system engineering.

Definition of unacceptable accidents

An accident is an unplanned or undesired event that result in a loss of human, human injury, property damage, environmental pollution, mission loss, etc.

Assigning a level of severity to accidents is recommended by standards, for example CENELEC standards, in order to prioritise the actions and effort necessary to eliminate or mitigate hazards related with unacceptable accidents.

Definition of the system boundaries

Boundaries determine which conditions related to accidents are considered part of the system and which are considered part of the environment.

Boundaries should be drawn to include conditions related to accidents over which the system designer has some control.

Identification of high-level system hazards

A hazard is a system state or a set of conditions that, together with a particular set of worst-case environmental conditions, will lead to an accident.

A small number of system-level hazards should be identified first. The system-level hazards are refined during the design process when the design alternatives are examined.

In general, the safety policy is that hazards leading to human loss or injury must be eliminated.

Definition of system safety constraints

System safety constraints are the conditions the system itself, its organisation and its development process must fulfil to prevent hazards from occurring.

System-level constraints are broken down and allocated to each system component during the system engineering process. The process iterates over the components as they are refined and decomposed and as design decisions are made.

STPA uses the system functional control diagram, the system-level and component-level hazards and the corresponding system-level and component-level constraints.

STPA process has two steps: 1) Identify the potential inadequate control actions that can lead to a hazardous state; 2) Determine how each potentially hazardous control action identified in step 1 can occur.

The first step is a “what-if” analysis of the control actions according to the four causes of accidents mentioned in the previous section. For instance: what consequences if the examined safety control action is not provided? What consequences if the opposite of a safe control action is provided?

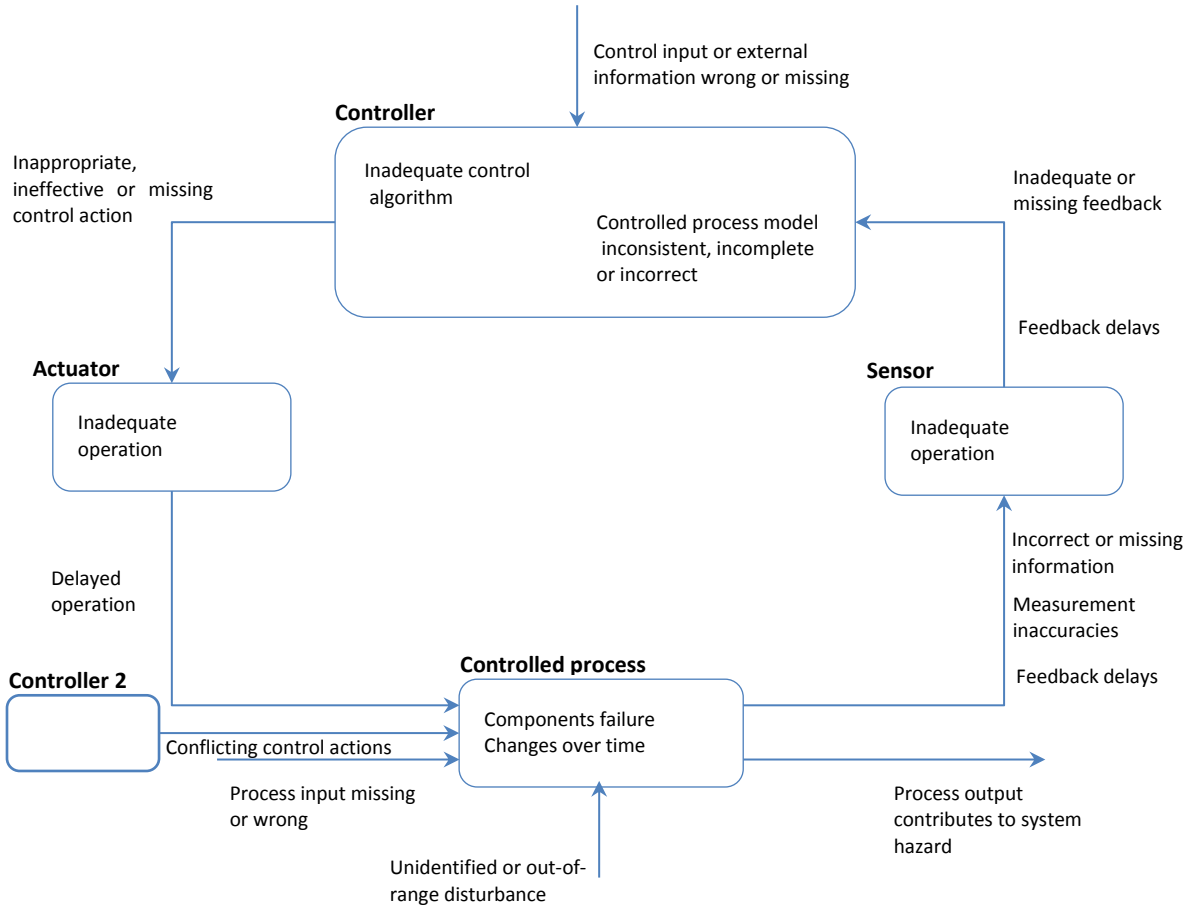

Figure 2: A classification of control flaws leading to hazards

The second step examines for each hazardous control action the parts of the control loop that can cause it and considers how the control actions can degrade over time and the protections that can be built. The analysis is achieved following the general accident causal factor schema of Figure 2 extracted from [11].

The causes of accidents are divided in three categories: inappropriate controller operation, inappropriate behaviour of actuators and/or controlled processes, and flaws in communication and synchronisation among controllers and decision makers.

Inappropriate controller operation may be caused by wrong (missing) controls or inputs provided (not provided) by higher level controller, by a control algorithm incorrectly specified or implemented or by an incorrect representation of the controlled process in the controller process.

Inappropriate behaviour of actuators or controlled processes may result from flaws in the transmission of control commands, from failures or faults in these components, or from missing or wrong inputs from other system components.

Finally, accidents are most likely in overlap areas or boundary areas where two or more controllers of the same process did not coordinate appropriately their control actions or the coordination did not happen effectively.

3.3 STAMP and STPA in the railway domain case study

Using STAMP and STPA in the case study was not an absolute necessity given that we have at our disposal the hazard analyses made by Alstom for the IXLs it has commissioned and certified in the past. Our motivation was rather to assess the improvements that these methods can bring to current

practices in the railway industry. The following sections summarise the work done for the hazard analysis of the IXL-DC.

3.3.1 Identification of accidents, system-level hazards and constraints

According to N. G. Leveson, the first step of the hazard analysis is the identification of the possible accidents of the system, of the system-level hazards leading to them and of the constraints on the system that prevent or mitigate these hazards. This work is independent of STAMP and STPA and is part of a standard preliminary hazard analysis.

Table 1 contains the accidents, and the system-level hazards and constraints of a signalling system involving the ATS, ATC and IXL components; this information is extracted from the preliminary hazard analysis of an operational Alstom signalling system.

A1	**Rear collision**		
	H1.1	The distance between two successive trains is less than the braking distance of the follower train.	
		C1.1	The system shall maintain in front of each train a track section free of obstacles longer than the braking distance of the train.
		C1.2	The system shall prevent trains from running backwards.
A2	**Side collision**		
	H2.1	The distance between a train running on a route which crosses the route of another train and the trajectory of the latter train is less than the braking distance of the former train.	
		C2.1 = C1.1	The system shall maintain in front of each train a track section free of obstacles longer than the braking distance of the train.
		C2.2	The system shall not authorise simultaneously routes that intersect.
A3	**Head-to-head collision**		
	H3.1	Two trains run on the same track in opposite directions.	
		C3.1 = C1.1	The system shall maintain in front of each train a track section free of obstacles longer than the braking distance of the train.
		C3.2	The system shall not authorise simultaneously opposite routes that overlap or end in the same place.
A4	**Collision with objects on the track**		
	H4.1	A hazardous object fell or has been left on the track.	
		C4.1	Maintenance procedures must ensure that no hazardous object is left on the track after a maintenance operation.
		C4.2	Operation procedures must ensure that no hazardous object is on the track during train operation.
A5	**Collision with system structure**		
	H5.1	The distance between a train and the end of line buffer is less than he braking distance of the train.	
		C5.1 = C1.1	The system shall maintain in front of each train a track section free of obstacles longer than the braking distance of the train.
	H5.2	Signalling system equipment is misplaced.	

		C5.2	Commissioning and maintenance must ensure that signalling equipment is out of reach of trains.
A6	**Derailment due to the instability of the train**		
	H6.1	A train runs at excessive speed relative to the configuration or the structure of the track.	
		C6.1	The system shall prevent trains from exceeding the maximum speed authorised by the configuration or the structure of the track sections.
A7	**Derailment due to a loss of guidance**		
	H7.1	A train runs on a point locked in the wrong position.	
		C7.1	The system shall lock points in front of a train in the position required by the planned route of the train.
	H7.2	A train runs on an unlocked point.	
		C7.2	The system shall ensure that points are locked in front of an approaching train or under a train.
	H7.3	A rail is damaged.	
		C7.3	Commissioning and maintenance must ensure that rails are safe.
	H7.4	A hurtful object fell or has been left on the track	
		C7.4 = C4.1	Maintenance procedures must ensure that no hazardous object is left on the track after a maintenance operation.
		C7.5 = C4.2	Operation procedures must ensure that no object is on the track during train operation.
A8	**Persons on track struck by a train**		
	H8.1	Maintenance workers are on a non-protected track maintenance zone.	
		C8.1	The system must protect track maintenance zones.
	H8.2	Passengers are on a non-protected track evacuation zone.	
		C8.2	The system must protect track evacuation zones.

Table 1 : Accidents, System-level Hazards and Constraints

In the case study we dealt only with collisions and derailments caused, directly or indirectly, by IXL.

3.3.2 Control structure of an interlocking

The second step of the hazard analysis is the creation of the control structure of the signalling system prescribed by STAMP. Six processes are involved: the traffic operator, the ATS, the ATC, the IXL, the trains and the trackside equipment.

Figure 3 represents the control structure of the studied system.

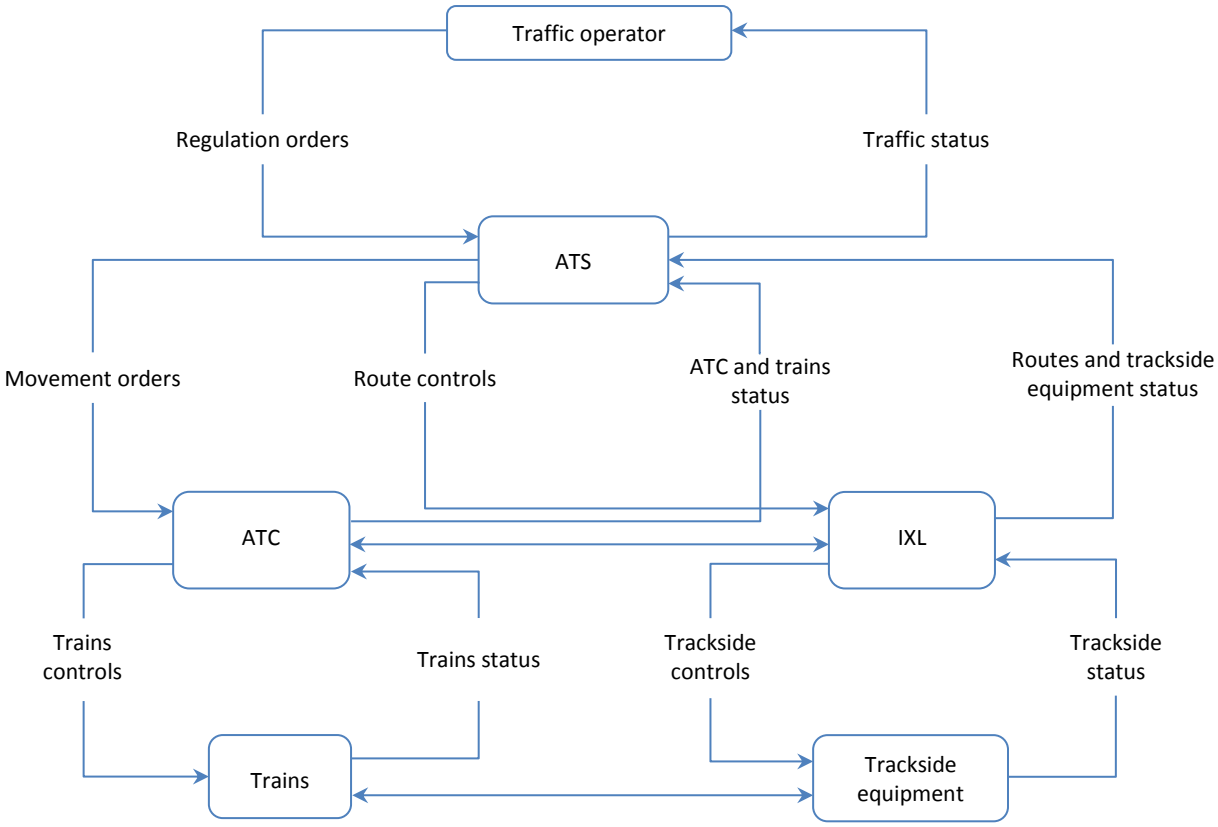

Figure 3: Control structure of a railway signalling system

The traffic operator controls the ATS with traffic regulation orders and receives as feedback information on the traffic status. Regulation orders include assignment of routes and time-tables to trains and destruction of routes. The traffic status includes the status of trains and of the ATC and the status of trackside equipment.

The ATS controls the ATC with movement orders and receives as feedback information the status of trains and of the ATC. Movement orders include destination information, departure and arrival times and possible tolerances. The status of the ATC includes the operation mode, the location of the train and its travel direction. The status of trains includes their speed and the status of their doors.

The ATS controls also the IXL with route controls and receives as feedback information the status of the routes and of the trackside equipment. Route controls include the creation/destruction of routes and the blocking/unblocking of points or signals. The status of routes and trackside equipment includes route set/unset status, the position of points, the aspect of signals and the occupancy of the track.

The ATC controls the trains with train controls and receives as feedback information the status of trains. Train controls include traction and brake commands and opening/closing door commands. The status of trains includes their speed and the status of their doors.

ATC provides IXL with the track sections occupied by the protection envelopes of trains.

The IXL controls the trackside equipment with device controls and receives as feedback information the status of this equipment. Device controls include movement of points and lighting of signals. The status of the trackside equipment includes the position of points, the aspect of signals and the occupancy of the track.

IXL provides ATC with the authorised traffic directions on the sections of the track.

Trains control trackside equipment, for instance track circuits, axle counters or beacons, and in turn trackside equipment, typically points, controls trains.

3.4 Hazardous control analysis

STPA starts after preliminary hazard analysis and control structure construction. Its first step is the identification of hazardous controls. For each control of the control structure, the impact on the system of the four causes of accidents considered by STAMP is analysed.

Table 2 presents the analysis of some controls and outputs issued by the IXL. The first column identifies the analysed control or output. The second column records the consequences of not providing a safe control. The third column records the consequences of providing an unsafe control (i.e. the control allows the controlled process to perform actions in a context where they will lead to hazards). The fourth column records the consequences of providing a safe control too early or too late or in a wrong order. The fifth column records the consequences of stopping a safe control too soon or applying it too long.

"Set signal permissive" is a control that allows a train to enter a safe route protected by a signal. "Set signal restrictive" is a control that prevents trains from entering an unsafe route protected by the signal. "Control point" is a control that moves a point to a desired position. "Traffic direction" is an output from the IXL to the ATC that authorises the train to run in the given direction. "Track circuit occupancy" is an output from the IXL to the ATC indicating the occupancy state of a section of track.

Control/ Output	**Not providing causes hazard**	**Providing causes hazard**	**Wrong timing/order causes hazard**	**Stopped too soon/applied too long causes hazard**
Set signal permissive	Not hazardous	The distance between a train and an in-front obstacle is less than the braking distance of the train (H1.1, H2.1, H3.1); A train runs on a point locked in an incorrect position or on an unlocked point (H7.1, H7.2); A train runs at an excessive speed (H6.1)	Too early: cf. 2nd column	Too soon: Not hazardous
			Too late: Not hazardous	Too long: cf. 2nd column
			Wrong order : Not applicable	
Set signal restrictive	The distance between a train and an in front obstacle is less than the braking distance of the train (H1.1, H2.1, H3.1); A train runs on an unlocked point or on a point locked in an incorrect position (H7.1, H7.2); A train runs at an excessive speed (H6.1)	Not hazardous	Too early: Not hazardous	Too soon : cf. 2nd column
			Too late : cf. 2nd column	Too long : not hazardous
			Wrong order : Not applicable	

Control/ Output	Not providing causes hazard	Providing causes hazard	Wrong timing/order causes hazard	Stopped too soon/applied too long causes hazard
Control point	A train runs on a point locked in an incorrect position (H7.1)	A train runs on a point locked in an incorrect position or on an unlocked point (H7.1, H7.2)	Too early : A train runs on an unlocked point (H7.1)	Too soon: A train runs on an unlocked point (H7.1)
			Too late: A train runs on an unlocked point (H7.1)	Too long: Not hazardous
			Wrong order: Not applicable	
Traffic direction	Not hazardous	Two trains run on the same track in opposite directions (H3.1)	Too early : cf. 2nd column	Too soon: The distance between a train and an in front obstacle is less than the braking distance of the train (H1.1, H2.1, H3.1); A train runs on an unlocked point or on a point locked in an incorrect position (H7.1, H7.2); A train runs at an excessive speed (H6.1)
			Too late: Not hazardous	Too long: Not hazardous
			Wrong order : Not applicable	
Track circuit occupancy	The distance between a train and an in front obstacle is less than the braking distance of the train (H1.1, H2.1, H3.1)	Not hazardous	Too early : cf. 1st column	Too soon : cf. 1st column
			Too late : cf. 1st column	Too long : cf. 1st column
			Wrong order: cf. 1st column	

Table 2 : Hazardous Controls Analysis

From the precedent analysis it is possible to draw constraints on the interlocking system.

IXL.C.1	IXL shall consider a route safe and locked if all the points involved in the route (including conjugated and protection points) are positioned as required by the route and locked; if all the sections of the route have a traffic direction identical to the direction of the route; if the route does not intersect or overlap any other route; and if the route is not juxtaposed to a route in the opposite direction.
IXL.C.2	IXL shall control to permissive a restrictive signal only if a route downstream of the signal is safe and locked.

IXL.C.3	IXL shall control to restrictive a permissive signal if no route downstream of the signal is safe and locked.
IXL.C.4	IXL shall control and lock a point in the position required by the route of a train.
IXL.C.5	IXL shall control a point only if no train is close or on the point.
IXL.C.6	IXL shall transmit traffic directions corresponding to route directions.
IXL.C.7	IXL shall not remove the traffic directions in front of a train traveling a route.
IXL.C.8	IXL shall transmit the track occupancy information provided by the sensors on or along the tracks as soon as, and in the order, it has received them.

Table 3 : Interlocking constraints

The first constraint brings together the conditions on routes that prevent the hazards identified in Table 2. The condition on the position and locking of points prevents side collisions due to intersecting or too close routes, and derailments due to train instability or inadequate speed. The conditions on traffic directions and routes configuration prevent head-to-head and side collisions.

The other constraints of table 3 result from the hazardous controls analysis of table 2. The second constraint prevents the hazards caused by providing a wrong permissive signal control. The third constraint prevents the hazard caused by not providing a correct restrictive signal control. The fourth constraint prevents the hazards caused by not providing a correct point position control or providing an incorrect point position control. The fifth constraint prevents the hazard caused by providing too soon or too early a point position control. And so on.

3.5 Casual scenarios analysis

This section presents two scenarios leading to hazardous controls that illustrate the use of the accident causal factor model of STPA presented in section 3.2.

The first scenario, presented in a graphical form in Figure 4, shows that an incorrect implementation may lead the interlocking to issue a hazardous “Signal permissive” control. Following Figure 2: in section 3.2, the lists inside the IXL and ATC boxes indicate the possible causes of incorrect controls.

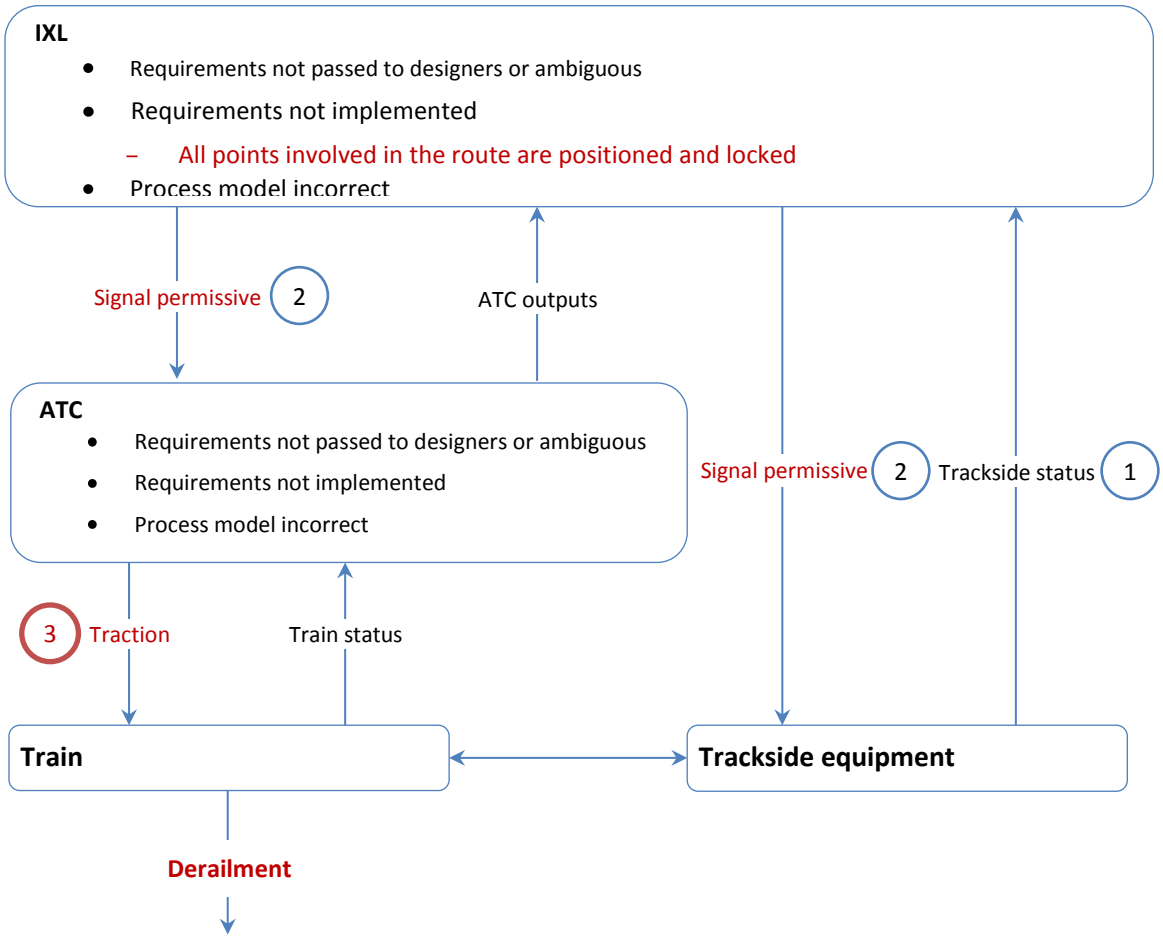

Figure 4: Unsafe signal permissive scenario

The red legends indicate the flaws of the present scenario that lead to an accident and the circled numbers indicate the order in which the flaws occur. In this scenario the trackside equipment provides correct feedback information to IXL (①); the IXL because either its control algorithm or its instantiation are incorrect ignored a point when it set a route and provides a "Signal permissive" control to ATC and trackside equipment while the considered point is not in the correct position or unlocked (②); then ATC calculates a too long movement authority and applies traction to the train (③) which may run on the hazardous point and derail.

Figure 5 illustrates a less trivial scenario that shows that poor detection of trains on the track combined with particular operation conditions may lead to an accident. A short train is mute either because it is not equipped with onboard ATC-CC (Carborne Controller) – typically a locomotive – or because the data communication system (DCS) between its ATC-CC and the trackside ATC-ZC (Zone Controller) failed (①); nevertheless, the ATC-ZC creates a protection envelop around this train because it is detected by track circuits; the train runs slowly around the joint between two consecutive track circuits (where train detection is faulty) and is no longer detected (②) – in nominal conditions at least one track circuit should have detected the train if it wasn't too short; the trackside equipment transmits this information to the IXL (③); the IXL informs the ATC-ZC that the track sections under the train are free (④); the ATC-ZC removes the protection envelop around the mute train because it is totally over free track circuits and calculates the movement authority for the other trains ignoring the presence of the mute train (⑤) which may collide with non-mute trains.

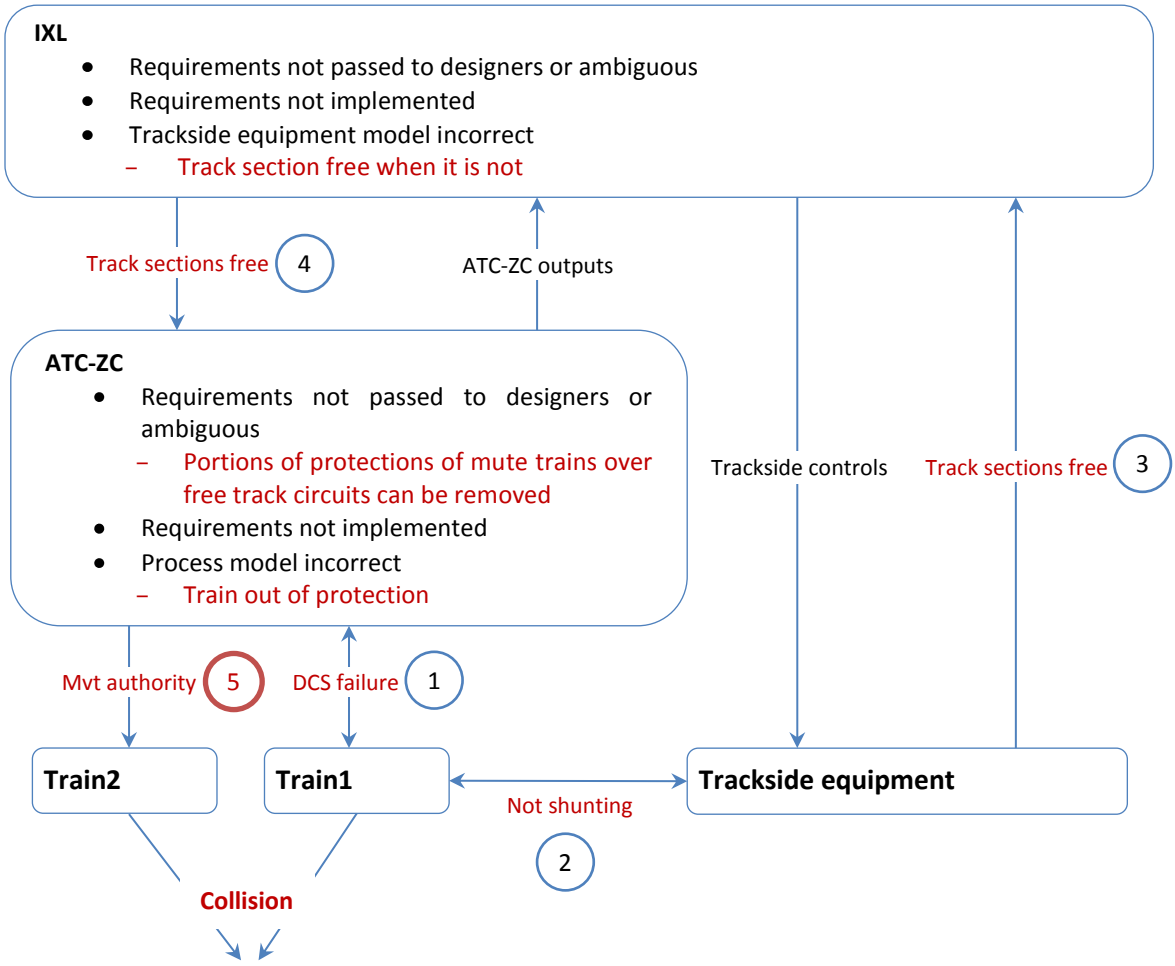

Figure 5: Unsafe track occupancy scenario

3.6 Evaluation

Alstom and almost all the railway industry use the conventional hazard analysis techniques recommended by the applicable standards (CENELEC, IEEE, …): Fault Tree Analysis (FTA), Failure Mode Effects Analysis (FMEA), Failure Mode Effects and Criticality Analysis (FMECA), Markov techniques, etc. N. Leveson already explains in her book the benefits of STAMP/STPA compared with these techniques. The case study allows us to assess these claims.

The first notable difference between STAMP/STPA and our standard process is the integration of the regulatory and organisational agents in the system hazard analysis. This increases the scope of the analysis and allows considering more interaction between the analysed system and its environment. Thus, as claimed N. Leveson, hazard analysis with STAMP/STPA is more comprehensive than with our conventional methods.

The second notable difference between STAMP/STPA and our standard process is that the STAMP/STPA hazard analysis is based on a control model of the system while our standard hazard analysis is based on a functional model of the system. The case study showed that control structure as modelling paradigm allows an abstract and relevant description of the system that highlights the roles and the responsibilities of the agents in maintaining safety. Furthermore, we believe that the control structure paradigm is more effective than the function paradigm for hazard analysis because it allows linking more easily and directly the causes and effects of hazards. Thus hazard analysis with STAMP/STPA is more explicit and shorter than with our conventional techniques.

However, despite its technical qualities, the dissemination of STAMP/STPA in the railway industry needs academic and commercial ecosystem which is lacking today. More training courses, support tools and documented success stories are necessary to overcome the habits of the safety community of railway industry.

4. Development of the Interlocking dynamic controller Event-B model

4.1 Model construction

The model of the IXL-DC was constructed in two main phases. The first phase constructed by successive refinement steps a single component Event-B model that models the relevant aspects of the complete signalling system. This component formalises the system-level safety requirements and the system-level events dealing with train movements and protection. The starting point of construction is an initial model that gives a very abstract view of the system. This model is completed gradually following a refinement plan that defines the number of refinement steps to be done and the concepts and events that each step introduces. The proof of the last refinement, i.e. the system-level Event-B model, ensures that system-level safety requirements are met.

The second phase of the construction decomposed the system-level Event-B model into three Event-B models: the Event-B model of the IXL-DC, the Event-B model of the environment, i.e. all the aspects of the ATS, ATC and trackside equipment relevant for the IXL-DC, and the Event-B model of the communications between the IXL-DC and the environment. The Event-B model of the environment assumes that ATC avoids rear collisions and overspeed, and that ATS unlocks signals and points in safe conditions. The model of the IXL-DC avoids head-to-head, side collisions, and derailment due to wrong point position or movement. The proof of decomposition provides evidence that, even if the IXL is unsafe, the IXL-DC ensures compliance with system–level safety requirements.

The first part of this section presents the refinement plan of the System-level Event-B model. The second parte presents the decomposition of the system-level Event-B model.

4.1.1 Refinement plan of the system-level Event-B model

4.1.1.1 Overview

The system-level Event-B model involves the basic IXL commands, the environment and IXL-DC. The starting point of construction, the abstract level, describes a high-level view of the whole system. As the refinement goes further, more accurate views of the system are given. The IXL-DC plays the role of a filter of IXL's outputs to environment. These include commands of aspects of signals, of movements of points, of directions of blocks and of locks of overlaps. In reverse, the IXL-DC receives the changes of the environment status. Changes are modelled by events. These include train movement (train moving backward or forward, entering or leaving the considered track network, turning around, immobilising), train shunting or deshunting a track circuit, point movement and signal aspect changes. Figure 6 shows the flow of information between the IXL, the IXL-DC and the environment.

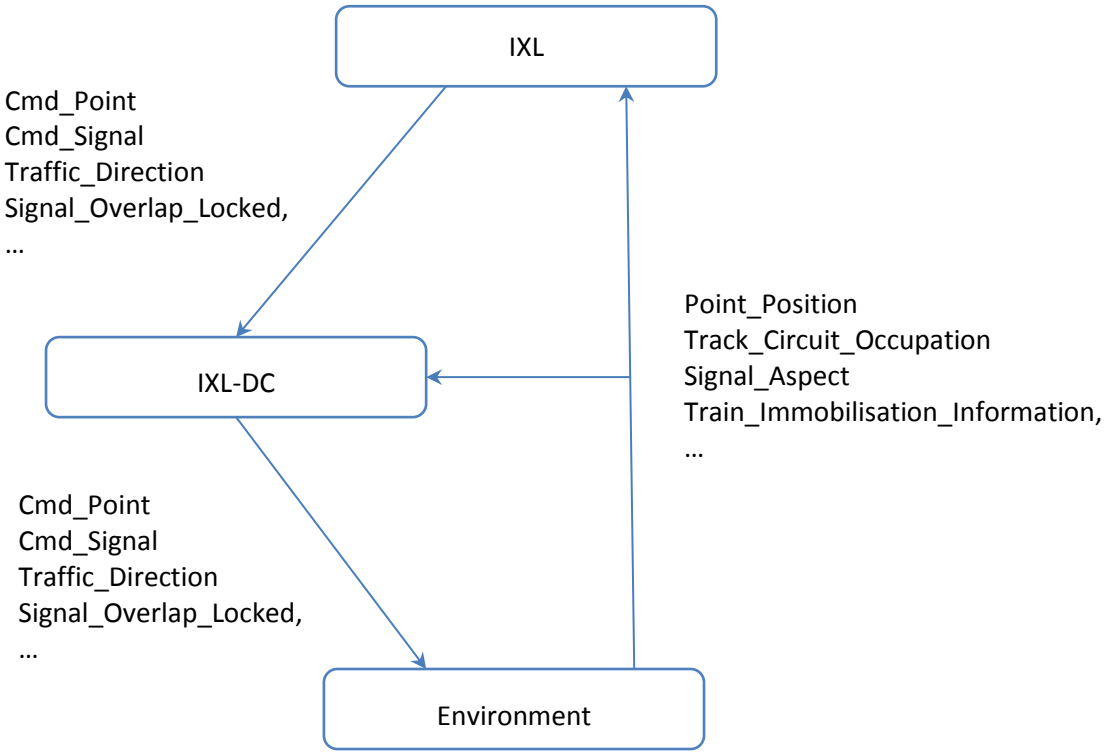

Figure 6: Information flow between IXL, IXL-DC and environment

IXL-DC ensures that commands sent to environment are safe. More precisely:

1. No command of points on which a train can potentially move.
2. No command of signals leading to an incompatible itinerary.
3. No incompatible traffic direction or signal overlap locked information.

These criteria protect the system against:

SAFE_1: Train derailment

SAFE_2: Train collisions

The following descriptions are extracted from [3].

4.1.1.2 The initial model

The initial model gives an abstract representation of safe train movements. In this first model, trains on the track network are modelled by chains of oriented blocks and the non-collision properties are expressed by the compatibility of these areas. The invariants of the model are:

- (0_1) Points under a train must be well positioned.
- (0_2) Blocks occupied by two different trains are compatible (see below).
- (0_3) Trains leave the network at its extremities.

Compatible oriented blocks are defined as follows: two oriented blocks are compatible if they are different, if they are not opposite blocks of the same underlying non-oriented block and if their underlying non-oriented blocks are not in a fouling area of a point.

The invariants 0_1 and 0_2 refine SAFE_1 and SAFE_2, respectively. 0_3 ensures that a train cannot "disappear" on the track network.

Modelling context data: This initial model uses the following data of track network configuration:

- Static and dynamic chaining of blocks;
- Configuration of fouling blocks. Note that two oriented blocks are incompatible if they are either fouling blocks or opposite blocks. A block is incompatible with itself.
- Turn around blocks.

Events:

The events of this refinement model basic movement of trains (including turnaround), movements of points and lighting of signals. For a full description see [3].

4.1.1.3 1st Refinement: Area reservation

In this refinement, protected areas are introduced. The notion of protected area is more general than that of train movement authorization zone: a protected area is not necessarily assigned to a train. Protected areas must ensure non-collision and non-derailment of trains. Safety properties can be expressed as follows:

- (1_1) A train is covered by a protected area
- (1_2) Points in a protected area must be well positioned
- (1_3) Protected areas must be pairwise compatible

The property *(1_1)* is a link property. As trains are covered by protected area, properties (0_1) and (0_2) in the abstract machine can be refined into the properties (1_2) and (1_3). In other words, the abstract properties hold whenever the concrete properties hold.

Protected areas can be implemented by chains of oriented blocks on the dynamic track network.

Modelling context data: this refinement uses the same modelling context data as that of its abstract level.

Events: This refinement introduces events that manage protected areas: create/remove/ extend/reduce/split protected area. Events of the previous model are refined to take into account protected areas. For a full description see [3].

4.1.1.4 2st Refinement: Filtering point commands

This refinement introduces the filtered commands of points into the model and refines the movement of points. The movement of a point from left (right) position to unknown position implies that its conjugated points are also moved to unknown position. A point is in moving from unknown to left (right) position implies that its conjugated points are in moving from unknown position to their corresponding positions. A point is moved from unknown position to left (right) if it was moving to left (right). A point moves only when commanded. Safety properties are refined as follows:

- (2_0) Point commands must not be contradictory (point is commanded in only one position). Furthermore conjugated points of commanded points must not be contradictory (a point is associated to only one position to move to).
- (2_1) A commanded point and its conjugated point must NOT belong to a protected area.

When the properties *2_0* and *2_1* are verified, properties *1_2* and *1_3* are also verified.

Modelling context data: In addition to context data used in the abstract level, this refinement uses the following context data:

- Conjugated point of a point

Events: This refinement introduces events modelling command of points and transient states of points. Events of the previous model are refined to take into account commanded and not yet controlled points. For a full description see [3].

4.1.1.5 3rd Refinement: Traffic directions

This refinement introduces traffic direction commands and all booked blocks. All booked blocks include booked blocks and overlap booked blocks. A traffic direction can be set on a block, only if the block was already booked and had no direction. In our model, we examine booked blocks from the moment directions have first been set on them. Therefore points in point commands or in their conjugated points have no underlying booked blocks. The refinement is based on the following safety properties:

- (3_1) Traffic direction is defined only on a booked block.
- (3_2) Protected areas are covered by booked blocks.
- (3_3) On all booked blocks, no point is commanded nor its conjugated point.
- (3_4) All booked blocks are compatible.

The properties *(2_0)* and *(2_1)* hold when *(3_2), (3_3)* and *(3_4)* hold.

Modelling context data: this refinement uses the same modelling context data as that of its abstract level.

Events: This refinement introduces events that manage traffic directions and blocks of overlaps: command/set/release/reverse traffic direction; book/unbook overlap block. Events of the previous model are refined to take into account traffic directions. For a full description see [3].

4.1.1.6 4th Refinement: Track circuits

This refinement introduces track circuits in order to strengthen guards of events. The controller does not know anymore the position of trains, but has a fuzzier picture based on track circuits that are shunted by trains.

Modelling context data: In addition to context data used in the abstract level, this refinement uses the following context data:

- Underlying SDD of blocks.

Events: This refinement does not introduce any event. It refines the events of the previous events to take into account track circuits. For a full description see [3].

4.1.1.7 5th Refinement: Signals

This refinement introduces signals and related timers. In this machine, protected area formation is defined: when a signal turns to permissive aspect, a protected area is extended from the downstream block of the signal until the upstream block of the next signal or possibly sliding blocks of the next signal. Protected blocks are removed when blocks are un-booked.

Modelling context data: In addition to context data used in the abstract level, this refinement uses the following context data:

- Upstream and downstream of signals.
- Approach section of signals.
- Turnaround blocks of signals.
- Sliding sections of signals.
- Timers associated to signals.

Events: This refinement introduces events that manage aspects of signals according to timers and movement of trains: command_signalçrestrictive/release_signal_on_timer/release_signal_on_TORR. Events of the previous model are refined to take into account the introduced concepts. For a full description see [3].

4.1.1.8 6th Refinement: Interlocking commands

This refinement introduces IXL commands. Therefore, guards of previous filtered commands are strengthened by the presence of IXL commands.

Modelling context data: this refinement uses the same modelling context data as that of its abstract level.

Events: This refinement introduces events modelling IXL commands: command/no_command signal, command/no_command point, command/no_command traffic direction, book/unbook overlap. The events of the previous model are left unchanged. For a full description see [3].

4.1.2 Decomposition of the system-level Event-B model

This section presents the model decomposition activity carried out during the third period of the project.

Typically a control system consists of a collection of physical devices to be controlled according to certain operational and safety rules. ADVANCE Deliverable 5.3 on Process Integration describes a method for refining and decomposing a high-level model of such a system into separate models covering:

- the devices being controlled,
- the controller that makes the control decisions and
- the communications mechanisms connecting the controller to the devices.

A key feature of this approach is the stepwise decomposition of the model in order to manage the complexity of a system consisting of a number of different kinds of devices. In this section we outline the application of this decomposition method to the IXL-DC model of WP2. The high-level model of the IXL-DC consists of three major kinds of physical device:

1. Trackside light signals to the train drivers
2. Moving points that determine the connectivity between track sections
3. Trains that are positioned on track sections

Following the approach described in Deliverable 5.3, we decomposed the 6th refinement presented above (*sys_0*) in three major stages corresponding to the three kinds of physical device: First we extract the trackside signals, then the moving points and finally the trains. The diagram in Figure 7 provides an overview of this stepwise decomposition of the IXL-DC. Extracting a physical device model involves some preparatory refinement steps to introduce a controller variable representing the state of the device and to introduce explicit signalling (inter-component communications) between the controller and the device.

Considering Figure 7, the top of the hierarchy, *sys_0*, represents the high-level system model. The first refinement, *sys_0_1*, introduces the controller variable representing the trackside signal state as well as model elements representing the signalling between the controller and the trackside driver signals. The next step involves decomposing the sys_0_1 model into three separate models representing the physical trackside signals (*Device_Signals*), the signalling mechanism between the trackside signals and the controller (*Signal_Signals*) and the residual system model (*sys_1*). The residual system model is further refined and decomposed as illustrated in Figure 7 leading to the seven leaf component models shown in the figure. Three of these leaf models represent the three physical devices types (trackside signals, moving points and trains), three represent inter-component communications corresponding to the three device types and the final model, *IXL-DC* represents a model of the functionality required from a safe IXL software controller. Note that in this model decomposition work we are not distinguishing between an existing (unverified) IXL and the IXL-DC that filters unsafe IXL commands; our decomposed IXL-DC component specifies the behaviour from the combination of these.

From an engineering perspective, we assume that our goal is to provide a precise specification of the behaviour of the IXL controller *and* to demonstrate that the behaviour of the controller operating together with the physical devices and the communication mechanisms between the controller and the devices together conform to the high-level system properties capture in the abstract *sys_0* model. The model allows us to arrive at the controller specification in a constructive way starting with the abstract system-level model. The device (and device communications) models capture our assumptions about how the devices behave. Although our goal is not necessarily to implement the physical devices, we do need to make a judgement about how well the device models represent the known behaviour of the devices.

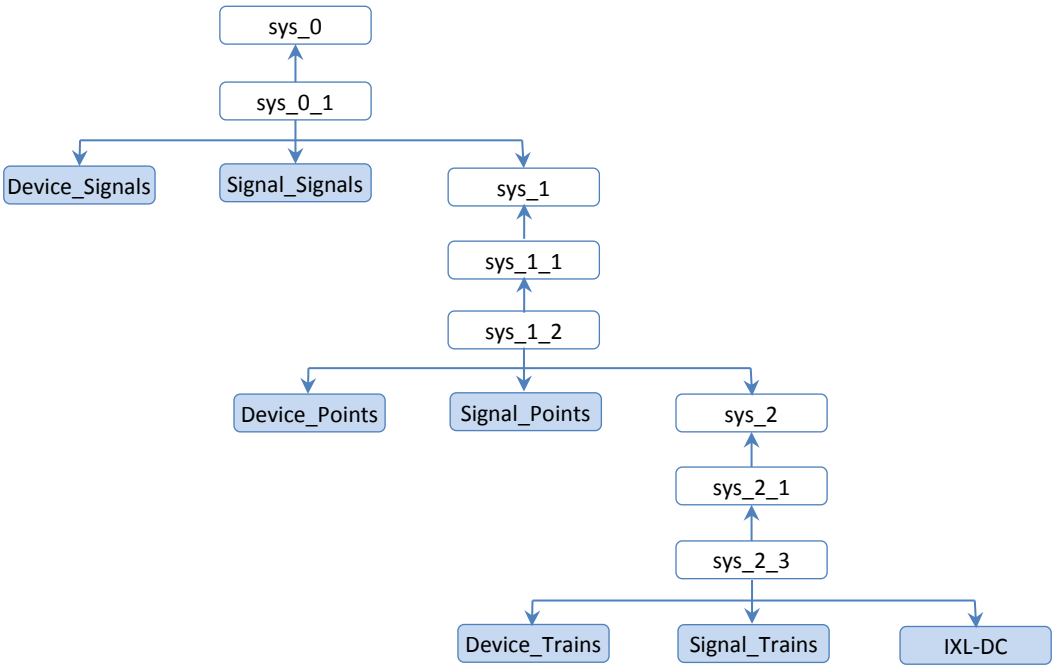

Figure 7: Overall decomposition of the system-level model

We now provide more details on the decomposition stages.

4.1.2.1 Extracting the trackside driver signals

The abstract model includes a variable *v_signals* representing the state of the trackside signals. In order to extract this from the main model, we introduce a variable representing the controllers version of this variable, *v_signalsC*. Following the decomposition guidelines, this controller variable should always be a safe replica of the device state. In this case, if the trackside signal is permissive (green), then the controller should know that, i.e.,

$$\forall\ s.v\_signals\ (s) = green \Rightarrow v\_signalsC\ (s) = green$$

It would be unsafe for the controller to believe that the signal was red when it was actually green as this would mean that the controller would believe that the signal was not permissive whereas a driver would see it as permissive. If the controller believes it is green when it is red, this is still safe as the train driver will see the signal as non-permissive. An important assumption about the signal is that they are constructed to be fail safe: if they are green, then it must be the case that the controller has commanded them to be green.

As well as introducing the controller version of the signal variable, we also introduce a variable modelling the actuation signal from the controller to the trackside signals and a variable representing a confirmation that the trackside device has enacted the actuation signal. Note that in many systems the confirmation will be in the form of a timer rather than an explicit confirmation signal from the

trackside. That is, the controller assumes that the actuation signal is enacted by a certain elapsed time.

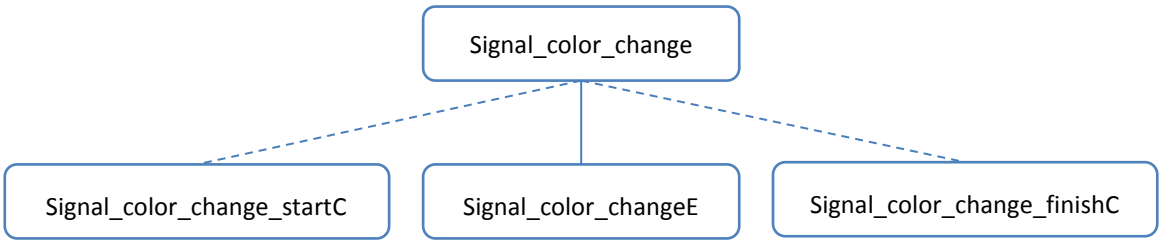

Figure 8: Refining the signal color change event

Following the decomposition guidelines we introduce new events to modify the variables introduced. The relationship between these new events and the existing signal event is shown in Figure 8. In the Event Refinement Diagram (ERS), the top event represents the event in the abstract model while the lower events represent events of the refinement model. The ordering of the refined events is significant: the left to right placement indicates that they get executed in left to right order. The dashed lines indicate new events while the solid line indicates that the lower event is a true refinement of the abstract event. We adopt the event naming convention from the guidelines, that is events whose name ends in 'C' represent controller events while events ending in 'E' represent events in the environment of the controller, i.e., in the physical device. Thus the sequence of events representing a colour change in the refined model are as follows:

1. *Signal_color_change_startC*: the controller sends an actuation signal for the colour change
2. *Signal_color_changeE*: the trackside enacts the colour change
3. *Signal_color_change_finishC*: the controller updates its copy of the signal variable to the new colour

The refined model now has enough structure (in terms of variables and events) to decompose the model into the three sub-models shown in the first decomposition stage of Figure 7.

4.1.2.2 Extracting the moving points

In order to perform the next decomposition stage, extracting the points, we introduce the controllers version of the points variable. The states of the physical points are represented by the state transition diagram in Figure 9. The decomposition guidelines distinguishes between *stable* states and *transitory* states for a physical device with a transition from a stable to a transitory state corresponding to the initiation of a physical transition (triggered by the controller) and a transition from a transitory to a stable state corresponding to the completion of a physical transition (acknowledged by the controller). In the case of the points, the *normal* and *reverse* states of Figure 9 correspond to stable states, while the *unknown* state corresponds to a transitory state. Following the guidelines, a controller version of the points variable is introduced. Those events representing the transition from *normal*/*reverse* to *unknown* are refined by two sequential events, a request from the controller to the points followed by the physical transition. Those events representing the transition from the unknown state to *normal*/*reverse* are refined by a sequence consisting of the physical transition followed by acknowledgement by the controller. These event refinements are illustrated in Figure 10.

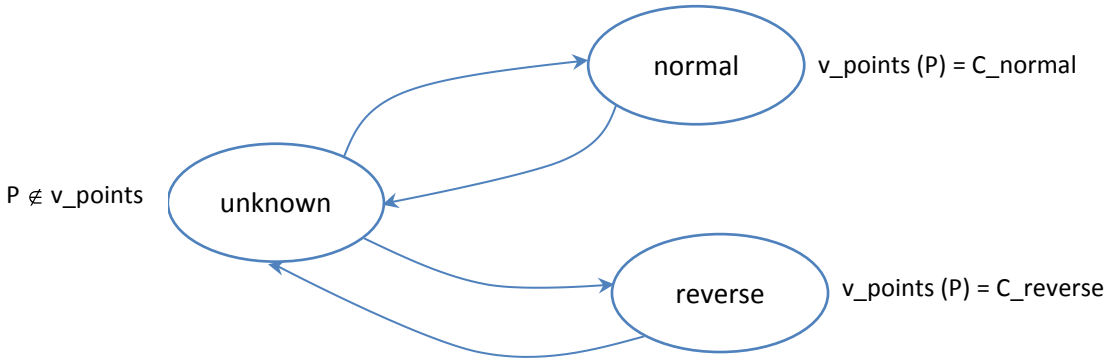

Figure 9: States and transitions of the points

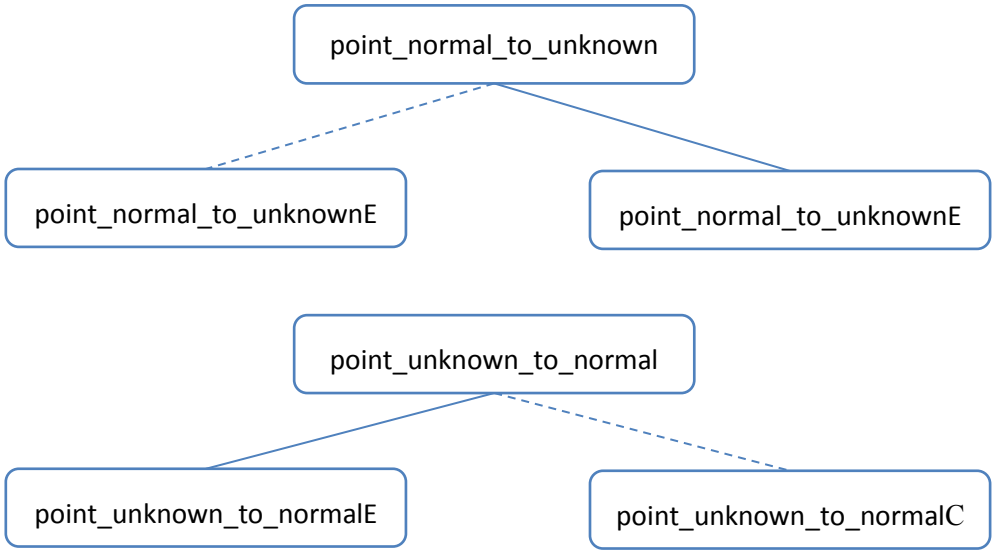

Figure 10: Refinement of points commands

The refinement just outlined is contained in *sys_1_1* of Figure 7. Prior to the decomposition, a further refinement is performed to introduce the explicit signalling between the controller and the points (*sys_1_2*).

4.1.2.3 Extracting the trains

The final decomposition stage involves extraction of the physical train positions from the residual model. This does not require the introduction of additional signalling from the controller to the trains since this is already provided by the trackside colour signals. The controller reserves blocks of tracks in advance for a train and these reserved blocks are a conservation estimate of the blocks actually occupied by a train. The controller doesn't need to know exactly which blocks are occupied by a train. As designers we need to ensure that the blocks occupied by a train are a subset of those blocks allocated to a train. The controller will only provide a permissive signal on a route when the blocks on that route have been allocated to a train. For safety purposes, the controller does not need to know when a train enters a block since, under the assumption that a train will only enter the block when it is permissive, the controller will already have allocated the block to the train. The controller does need to know when a train leaves a block so that the block can be freed. Thus we introduce

explicit signalling from the tracks to the controller to indicate when the train leaves a block. To do this, events corresponding to a train leaving a block are refinement to a sequence of two events, similar to the event refinement illustrated in the lower part of Figure 10. This event refinement is represented in model *sys_2_1* of Figure 7. The explicit signalling is then represented in model *sys_2_2* of Figure 7.

4.1.2.4 Evaluation

The decomposition guidelines of Deliverable 5.3 were largely suited to decomposition of the IXL model. One area where the decomposition differed is the signalling from the controller to the trains was already part of the abstract model. This reflects the strong conventional role that trackside signals play in ensuring safe operation in railways and also the property that trains are not directly controlled by the IXL but rather by a human operator. The IXL controls points and trackside signals and only monitors the occupancy of blocks by trains. The system level assumption is that human drivers will obey the signals. Prior to decomposition we represent this assumption by a guard on the signal colour for the events representing trains entering a permissive section of blocks. When decomposing the signals from the trains, this conjunction of guards (on the signals and on the train position) becomes a synchronisation between events of the signals model and the train model. There is scope for extending the decomposition guidelines to address this kind of assumption in the future.

4.2 Model validation

Part of the work described is this section, i.e. manual animation and initiation of automatic validation with validation test logs of actual signalling systems, has been done during the second period of the project and was reported in deliverable 1.3 ([3]). The other part, i.e. re-engineering of the architecture and tools of automatic validation and more detailed analysis of automatic animations and consequent rework on the IXL-DC model has been done during the third period of the project.

Validation of models is essential to ensure they define suitable systems. Validation of the model of IXL-DC allowed us to check that the IXL-DC can be integrated in a signalling system and is not too restrictive. In other words that IXL-DC checks the outputs of the IXL according to safety requirements without blocking them unduly and degrading the availability of the system. The proof of the model of the IXL-DC, presented in section §4.3, ensures that the IXL-DC is not too permissive according to safety requirements.

The validation of the model of the IXL-DC took place in two steps. The first step was made of manual animations of the model and the second step was made of automatic animations. Both steps used the ProB Animator and Model Checker tool developed by the University of Düsseldorf.

4.2.1 Manual animation

The model animated manually is the composite system-level model described above whose context modelling the static topology of the track and of the trackside equipment has been instantiated with data from a station of the Paris RER line B.

Figure 11 represents the architecture of the model. From bottom to top, the box TST-TRK represents the model defining static configuration data; boxes IXL-DC and IXL represent respectively the models defining the IXL-DC and the IXL; the box ENV represents the model of the environment that defines the relevant events of ATS, ATC and track-side equipment; the box DCS represents the data communication system that allow the exchange of information between IXL-DC, IXL and ENV. An arrow indicates that the component at its destination uses the component at its origin.

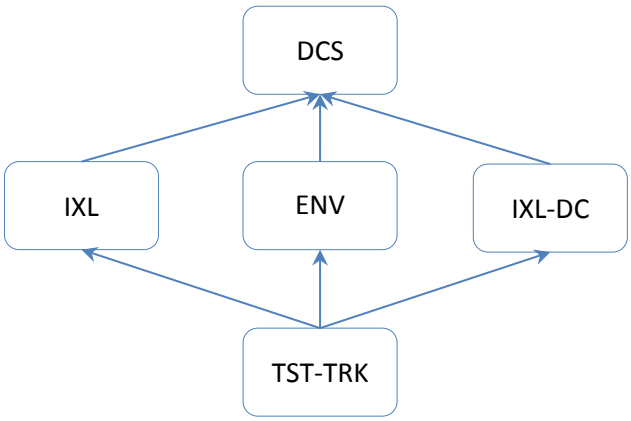

Figure 11: Architecture+ of the model animated manually

We used the facilities offered by ProB to display graphically the state of model during manual animation. Figure 12 shows the display.

Figure 12: Graphical display of a manual animation

There are four representations of the test track in the display. The first representation, the one on top of the display, presents the actual configuration of the track and the movement authorisations calculated by the ATC. The green lines represent the accessible track sections and the yellow lines represent the inaccessible branches of the points; the black lines represent the train movement authorisations. The small red and green boxes along the tracks represent respectively the restrictive and permissive signals. Signals are oriented. Signals placed above the lines control the movements of trains from left to right. Signals placed beneath the lines control the movements of trains from right

to left. Thus, this representation shows that all the points of the test track are on left position, that there are two movement authorisations, one from left to right on the top track covers all the track sections between two red (restrictive) signals and another one also from left to right that covers a single track section of the track beneath the previous one. All signals are restrictive.

The second representation presents the actual configuration and occupancy of the track. The green and yellow lines have the same meaning as above; the black lines represent the track sections occupied by the trains. Thus, this representation shows two trains running in the test track, one on the top track occupies the first two track sections of the limit of movement between the two red signals; the second occupies the first track section of the track beneath the previous one. All signals are restrictive.

The third representation presents the configuration of test track and the track occupancy as seen by the IXL-DC and the routes locked for trains. The green and yellow lines have the same meaning as above; the black lines represent the track sections occupied by trains; the read lines represent the locked routes and the figures in front of signals represent the value of timers related to routes. Notice that this representation shows that the IXL-DC is not yet aware of the presence of the second train in the first section of the second track – this track section is black in the first two representations but it is not in the third one. Notice also that the IXL-DC commanded green a signal but that the actual signal in the field is not yet green – it is red in the first two representations and green in the other two. The discrepancies between the actual configuration of the track and the configuration known by IXL-DC model the effects of communication delays between the trackside equipment and the IXL-DC.

The fourth and last representation presents the configuration of the test track as seen by the IXL-DC and the overlaps booked. The green and yellow lines have the same meaning as above; the read lines represent the locked overlaps and the figures in front of signals represent the value of timers related to overlaps. Thus, this figure shows that the overlap related to the signal that protects a point and that is the destination of a booked route is itself booked in order to protect the train from overtaking that signal.

The strength of manual animation is that it allows the creation and control of very detailed scenarios difficult to achieve otherwise. And indeed it was very helpful. It disclosed several errors in the models. It allowed analysing the behaviour of the IXL-DC in degraded operating conditions caused by track circuit, point or train shunting defaults. It also allowed analysing the consequences of the discrepancies between the actual configuration of the track and the configuration of the track known by the IXL-DC caused by communication delays between the trackside equipment and the IXL-DC.

4.2.2 Automatic animation

The weakness of manual animation of a model including several subsystems, each of which having many events, is that the manual scenarios may not be representative of the actual behaviour of the complete system. In other words it is difficult to be sure that the sequence of events triggered by a manual scenario corresponds effectively to a sequence of events of the actual system. For instance, it is difficult to reproduce manually the sequence of events of all subsystems that are triggered by a train running on a track. This is why we decided to animate automatically the IXL-DC model with scenarios that reproduce faithfully the environment where the IXL-DC is supposed to run.

As representative scenarios we chose the logs of validation tests done on Alstom's factory integration and validation platform (FIVP). This platform allows the testing of actual signalling systems in conditions close to actual operating conditions, notably with the description of the actual operation lines and with continuous models of the actual trains operated on these lines. A test log contains all the dated messages exchanged by the components of the signalling system during the test in the order they were sent, and represents, in some cases, several hours of operation during

which most of the operation situations occur. Thus, a test log contains all the information needed by the IXL-DC and reproduces faithfully the environment where the IXL-DC is supposed to run.

In order to automate the animation we created a scheduler component that includes the IXL-DC model and defines three events. The event that opens the file containing the test log and two mutually exclusive events: 1) an event that reads until exhaustion in the test log the messages exchanged during a constant period of time (400ms), that extracts from the messages the values of the inputs of the IXL-DC and that assigns these values to the appropriate variables of the IXL-DC; and 2) an event that fires until exhaustion all the events of the IXL-DC made fireable by the assignment of the input values.

Figure 13 presents the architecture of the automatic animation.

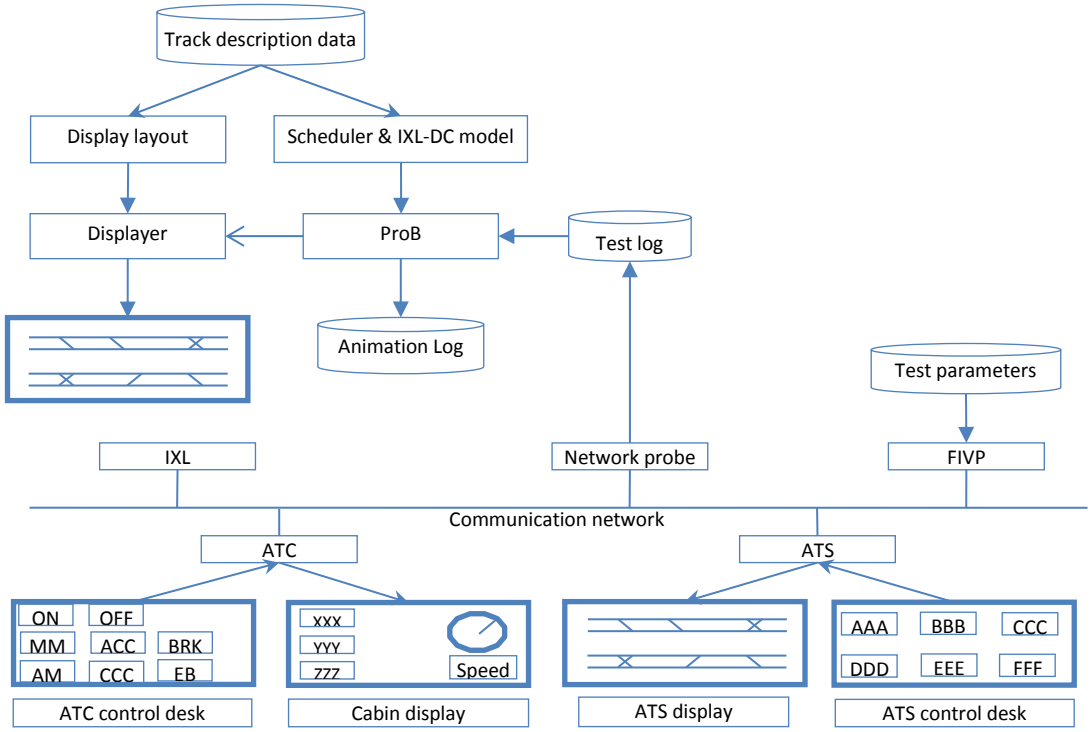

Figure 13: Automatic animation architecture

The bottom part of Figure 13 describes the architecture of the FIVP platform. The boxes IXL, ATC and ATS represent the actual components of the signalling system with their control desks and displays. The FIVP box represents the simulator involving the description of the test line, the continuous models of trains and the other test parameters. The box Network probe represents the tool that reads and logs the messages that transit in the communication network.

The top part of Figure 13 represents the architecture of the animation. The description of the test line is contained in a data base. The layout of the animation display and the instantiation of the IXL-DC model for the test line are drawn from this description. The ProB tool animates the scheduler that reads the test log and the instantiated IXL-DC model that checks the outputs of the IXL. It logs the triggered events and provides the displayer with IXL-DC's state information allowing it representing graphically the state of the line.

We used logs of validation tests of the signalling systems for urban networks of Malaga (Spain) and Santiago de Chile (Chile).

Figure 14 presents the animation display of the IXL-DC model with a test log of the signalling system for Malaga. Tracks are drawn from right to left. The two lines at the top right of the figure extend on

the left the two lines beneath, which, in turn, extend on the left the two lines beneath. The two lines at bottom right extend on the left the two lines above. Black lines represent the accessible sections of the tracks; yellow lines represent the inaccessible branches of points; blue lines represent occupied sections; red lines represent booked sections in DOWN direction (left to right in the figure); violet lines represent booked sections in UP direction; and rose lines represent booked overlaps. Red boxes represent restrictive signals; all other coloured boxes represent permissive signals.

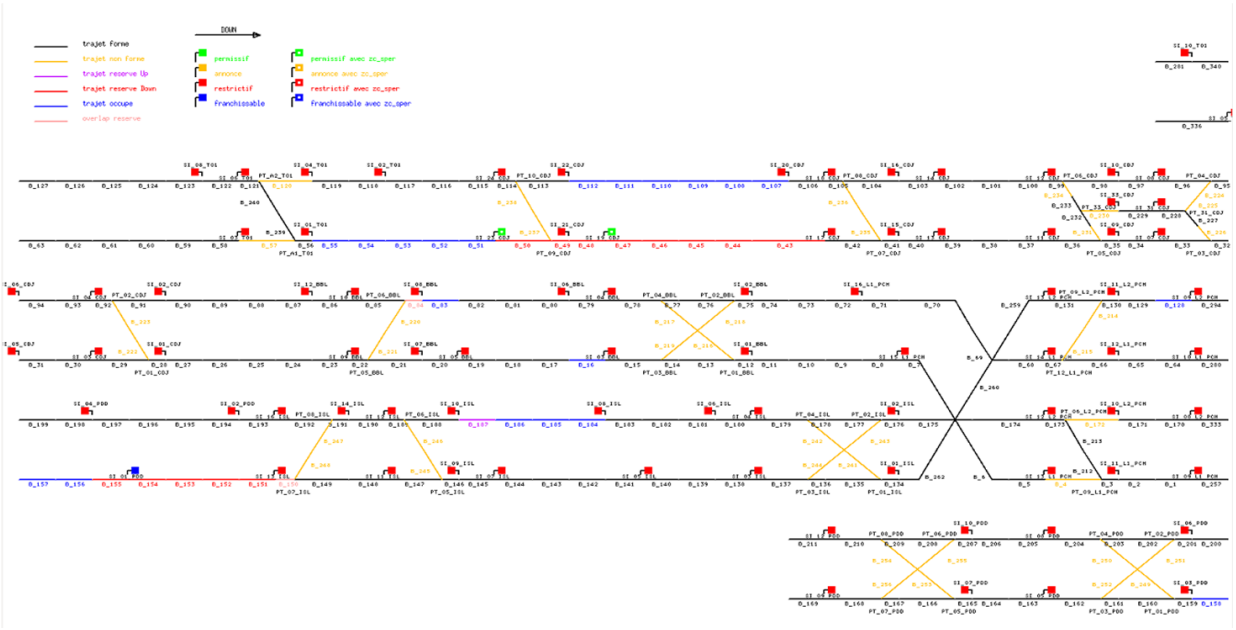

Figure 14: Graphical display of an automatic animation on Malaga network

We used three tests of the signalling system for Malaga. One test involves a single train that runs virtually all possible routes of the network. Its aim is to challenge the nominal behaviour of the system. Another test involves two trains running close behind one another on almost all the routes of the network. Its purpose is to test safety and availability of booking and releasing functions of routes and overlaps. The third test involves seven trains running on almost all routes of the network. Its purpose is also to test safety and availability of the system. This last test represents six hours of operation. Figure 14 is a screenshot of the animation display with the log of this last test.

Figure 15 presents the animation display of the IXL-DC model with a test log of the signalling system for Santiago de Chile. The representation of the tracks is similar to that of the tracks of Malaga presented above, except that UP direction is represented in this figure from left to right and that the two long lines at the top right extend on the left the two long lines beneath, which extend on the left the two long lines beneath and so on. In other words this figure represents two long parallel tracks.

We used one validation test of the signalling system for Santiago de Chile. It involves up to 24 trains that enter progressively revenue service tracks coming from depot (fourth and fifth lines at the top right of the figure). The figure above is a screenshot of this tests and shows six trains, two of which are waiting to enter revenue service tracks and four of which are already in revenue service.

Automatic animation was extremely useful to develop the model. It disclosed defects that cannot be disclosed by proof because they reflect incorrect comprehension of the functions of the IXL and of its dynamics. For instance automatic animation showed that the conditions on traffic directions for booking the routes and overlaps were too restrictive and blocked the system and that the order to trigger events that we originally implemented delayed unnecessarily the inputs and outputs of IXL.

We did not expect to find safety related defects in the IXLs with these tests because although not formally developed both systems have been thoroughly verified and validated.

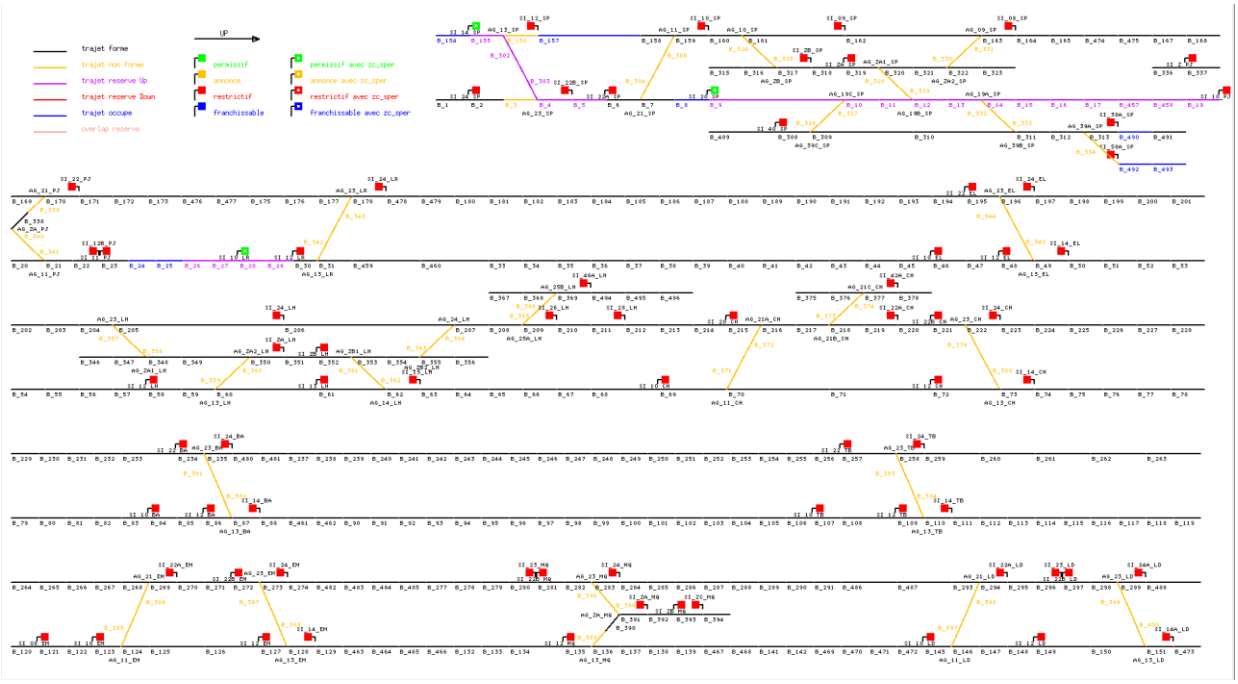

Figure 15: Graphical display of an automatic animation on Santiago de Chile network

4.3 Model verification

The verification of a formal model involves the critical review of the model, the proof of the model and the assessment of the proof. The purpose of critical review is to ensure that the model captures correctly and completely the requirements allocated to the modelled system, notably the safety related ones. Reviewers analyse the model and establish a correspondence between the concepts and the statements of the requirements and the data, invariants and events of the model. Reviewers also assess quality aspects of the model regarding economy of means, adequate usage of mathematical concepts and theories, and relevance of refinements. This analysis is important because these qualitative aspects in part determine the ability of the model to be proven more or less easily. Reviewers are members of the verification team which is independent of the design team that created the model. The University of Düsseldorf developed the ProR tool to manage the links between informal statement of requirements and their formalisation in the Event-B model, and assist reviewers.

The proof of the model discharges the proof obligations prescribed by the Event-B theory. Proof obligations are materialised by mathematical lemmas generated automatically by Rodin. Most of proof obligations are discharged automatically by the theorem provers of Rodin (AtelierB and SMT provers). The remaining proof obligations are discharged by the designers of the model that drive an interactive proof assistance. Interactive proof is a time-consuming activity that can be reduced if the appropriate techniques and methods that reduce the number of generated proof obligations are used to create the models.

One such technique is the *Theory* concept introduced by Event-B that allows defining an abstract set by the operators that create its elements, the operators that extract information from its elements and the properties that these operators fulfil. The use of a *Theory* reduces proof activity by the following mechanisms:

- Theorems: theorems allow splitting of a complex proof obligation. Moreover a theorem can be instantiated to be reused in different proof obligations.

- Proof rules: proof rules allow discharging several proof obligations matching a pattern. They can be defined to be used in interactive and/or automatic mode.
- Proof tactics: tactics allow a guidance of proof process according to the proof obligation form.

In the next paragraphs we first illustrate the use of the *Theory* concept in the IXL-DC modelling and then we illustrate the proof mechanisms mentioned above.

In the model of the IXL-DC we created a *Theory* called Graph to define the abstract data type that models the track layout, the routes and the trains. The following data types and operators are defined in the *Theory* Graph.

Graph: In the IXL-DC the static track layout is modelled by a binary relation from blocks to blocks. The dynamic track layout is modelled by an injective function from blocks to blocks that must be included in the relation modelling the static track layout.

Chain: In the IXL-DC, track occupation sequence plays an important role to filter IXL commands, including point commands, traffic direction commands, signal aspect commands and signal overlap commands. Track occupation sequences are modelled by the data type *chain* and its operators. A *chain* is a sequence of consecutive blocks. The operators on chains provide the following services:

- Inclusion of a chain in a graph: indicates whether a sequence of blocks is included in a graph of blocks.
- Extension of chain: add an element to the head/rear of a chain.
- Reduction of a chain: remove the head/rear of a chain.
- Chain turnaround: inverse the chain (head to rear, rear to head).
- Concatenation of two chains: create one chain from two chains.

The *Theory* Graph allows an abstract modelling of track occupation/reservation/liberation. The *Theory* Graph instantiates the *Theory* Closure which defines the standard closure operators.

Block incompatibility: IXL-DC must prevent from head-to-head and side collisions. The notion of "incompatible blocks" is used to designate fouling blocks of a point or opposite blocks of a same underlying track circuit. Additional operators are defined to provide the following services:

- Compatibility of chains: two chains are compatible if their blocks are not pair-wise incompatible.
- Compatibility between block and a chain: a block is compatible with all blocks of the chain.

Consistency of point positions: in the IXL-DC, a point must not be commanded to inconsistent positions. This filter includes the commands of conjugated points. For example, if the point pt_1 in normal position is conjugated with point pt_2 in reverse position (i.e. if pt_1 is in normal position, pt_2 must be in reverse position and conversely), it should not be possible to command the points pt_1 and pt_2 to normal position simultaneously.

- Consistency of positions of points: the positions of two points are consistent if the positions of all the points conjugated with these two points are consistent.

In addition to these operators, some utility operators are defined: Points on a chain, Chain between two signals, etc.

The theorems defined in the *Theory* Graph deal with chains and operators on chains. They are used in the interactive proof of:

- other theorems;
- proof rules;
- proof obligations of the IXL-DC model (machines and context components).

For instance, a theorem asserts that the result of the operator "*Chain turnaround"* which inverses a chain is also a chain. Another theorem asserts that there is no loop in a chain, i.e. no block is more than once in the sequence of blocks of the chain.

Theory operators are used to define invariants and events. In our model, proof rules are essentially used to discharge invariants. Thus, a proof rule generalizes a proof obligation of the form presented below where *V* and *V'* denote respectively the values of a variable before and after an event is fired, *INV(V)* denotes an invariant on the variable *V* and *GRD(V)* denotes the guard of the fired event.

```
   INV (V)
&  GRD (V)
=>
   INV (V')
```

For instance, the following proof rule aims to discharge proof obligations of events which extend a track occupation/reservation. It should be read as follows: if p_track is a function from blocks to blocks that represents a dynamic state of the track; and if p_ch represents one of the chains of blocks in p_track; and if p_elt represents a block that is not one of the blocks of p_ch; and if is_nxt_front (p_elt, p_ch, p_track) asserts that p_elt is the next block downwards the head block of p_ch in p_track; then the result of add_elt_to_front (p_ch, p_elt) which adds p_elt in front of p_ch is a chain of blocks of p_track.

The operator chain_on_graph is used in the IXL-DC model to model the "no derailment" safety property: a train is a sequence of consecutive blocks. This means that all the points under the train are well positioned and locked.

```
•r_ch_add_front   :   (both)      r_ch_add_front
 ■ p_track ∈ V ⇸ V
 ■ p_ch ∈ chain_on_graph(p_track)
 ■ ¬ is_elt(p_elt,p_ch)
 ■ is_nxt_front(p_elt, p_ch, p_track)
 ------------------------------------------------
 ■ add_elt_to_front(p_ch,p_elt) ∈ chain_on_graph(p_track)
```

Proof tactics used to prove IXL-DC model can be divided into three groups:

- Combination of default tactics: default tactics include AtelierB provers and SMT provers application.
- Combination of default tactics and proof rules related tactics: proof rule tactics include the application of rewrite rules (RbP0) or inference rules (RbP1). This combination allows applying *Theory* proof rules.
- Combination of default tactics and automatic definition expansion tactic: the automatic definition expansion tactic (RbPxd) is used to discharge well-definition proof obligations and proof obligations of initialisation events.

In addition to the *Theory* concept we used model refinement and decomposition, as presented in section 4.1, to simplify the model of the IXL-DC and to control the number and the complexity of proof obligations.

As seen above, designers create proof rules to help the provers of Rodin to discharge proof obligations. Some of these proof rules can be proved either automatically or interactively. The purpose of the proof assessment activity is to analyse the proof rules that could not be proved otherwise and provide evidence that they are indeed correct. Consequently, proof assessment

ensures that the complete the proof of the model is correct. Proof assessment is carried out by members of the verification team which is independent of the design team that proved the model.

4.4 Transition to software development

The development of a formal model of a system following the steps described in sections 4.1 to 4.3 improves the relevance, consistency and accuracy of the system specification and contributes to improve the implementation and validation of the system, whatever the implementation and validation methods employed. However, the model will be most valuable when used as a starting point for the development by stepwise refinement of formal models of subsystems and finally of software. It is particularly worthwhile to consider this use of system formal models when formal methods are already used to develop the software, which is the case of Alstom.

These are the reasons why we investigated in WP1 the transition from Event-B to Classical-B, the formal method used by Alstom Transport (France) for the development of ATC's safety critical software.

The process to move from a system model to a software model is quite obvious for Event-B and Classical-B practitioners. The system model is created progressively by stepwise refinement and finally it is decomposed according to the architecture of the system. Thus are created the Event-B system models of the subsystems of the initial system. This process of refinement and decomposition is iterated until are created the Event-B system models of the most elementary subsystems, those that will be implemented with hardware and software. The Event-B system model of an elementary subsystem is then refined until the software concepts are introduced. This ultimate Event-B model is then translated into Classical-B. We propose a translation is inspired by the work done to create the model of the IXL-DC animated automatically. The principles are the following.

An Event-B context is translated into a Classical-B machine that defines the same objects. This is straightforward because all the data types of Event-B can be easily translated into Classical-B data types and the predicate language is almost the same.

An Event-B machine is translated into two machines.

The first Classical-B machine defines the same variables than the Event-B machine and that for each event of the Event-B machine of the form:

```
EV =
ANY X WHERE
    G
THEN
    S
END
```

defines an operation of the form:

```
OP =
PRE
  ∃ X. G
THEN
   ANY X WHERE
      G
   THEN
      S
    END
END
```

And that for each event of the Event-B machine of the form:

```
EV =
SELECT G1 THEN
  S1
…
WHEN Gn THEN
   Sn
END
```

defines an operation of the form:

```
OP =
PRE
  G1 or … or Gn
THEN
  SELECT G1 THEN
    S1
  …
  WHEN Gn THEN
     Sn
  END
END
```

Although strictly speaking the operations are not refinements of the corresponding events, the translation corresponds to our intuitive comprehension of the behaviour of an event and to the behaviour implemented by ProB. That is to say, that the event is fired or animated only when its guard is effectively satisfied.

The second Classical-B machine created from an Event-B machine includes the first Classical-B machine and defines an operation that controls the activation of the operations of the latter that correspond to the events of the Event-B machine. In other words the second Classical-B machine schedules the operations of the first Classical-B machine.

5. Conclusions

We presented in the previous sections the work and the results of the railway domain case study of the ADVANCE project. We now assess the results of the case study as regarding its original objectives and the contributions of ADVANCE as regarding to system development. We terminate by presenting the future exploitation of the results of the case study and of ADVANCE.

5.1 Assessment of results of the case study

As regarding the fourth main objectives of the case study presented in the introduction we believe that we can state that they were reached.

First, we created, proved and tested using ADVANCE technology a formal model of an interlocking dynamic controller that enforces system-level safety requirements of an interlocking system. In terms of methods we used: Event-B and refinement and composition/decomposition techniques to design the Event-B model, the STAMP/STPA method to identify system-level safety requirements, and co-simulation to validate the model in realistic conditions. In terms of tools we used: Rodin and its plug-ins (i.e. theorem provers, “THEORY” and composition/decomposition) to create and prove the Event-B model, the ProB Animation and Model Checking tool to validate the model through manual and automatic simulation and ProR to record requirements.

Second, the interlocking dynamic controller we designed uses only the inputs and outputs of the controlled interlocking and thus is independent of its the implementation technology. Moreover, the interlocking dynamic controller is a generic component proved compliant with safety requirements

by means of mathematical proof based on the properties of the interlocked area rather than on a concrete and extensive description of it. The formal verification technique of the interlocking dynamic controller is therefore independent of the complexity of the interlocked area. Finally, the validation of the interlocking dynamic controller model with logs of factory tests of signalling systems involving an actual interlocking demonstrated the feasibility of the solution. Together these three facts show that we reached the second objective of the case study.

Third, we defined an industrial system development process by integrating ADVANCE technology in Alstom's system development process. The process is compliant with CENELEC EN50126 and EN50129 certification standards because Alstom's process is already compliant and because the integrated techniques are techniques recommended by the standards that support and formalise current practices. The process is inspired by Alstom's software development process and just like it takes full advantage of formal development in the sense that it avoids integration and validation tests covered by simulation and proof. The strategy for integrating the ADVANCE methods and tools into Alstom's CENELEC compliant development process is presented in ADVANCE Deliverable D1.4 [4].

Fourth, the case study provided valuable feedback to ADVANCE technology developers. It experimented thoroughly refinement and composition/decomposition techniques and contributed to improve Rodin and its "THEORY" and "composition/decomposition" plug-ins. Manual and automatic simulations of the model contributed to improve the performances of ProB, and its input/output and visualisation capabilities.

5.2 Contributions of ADVANCE to system development

ADVANCE proposes a development technology of Event-B system models based on, refinement and composition/decomposition techniques for the creation of the models, mathematical proof for the verification of the models, and simulation for the validation of the models.

Refinement and composition/decomposition are powerful techniques that allow progressive and flexible construction of models and simplify proof. But they must be used wisely. Too many refinements affect maintainability of the complete model because the modification of one level might imply the modification of several others. The case study showed that communication of independent systems can be modelled and managed using model decomposition techniques. We advocate that model decomposition should be made as soon as possible. Indeed, the separation of components (IXL-DC, IXL, environment and the communication) at an abstract level (w.r.t events/variables to be decomposed) makes it easier to have a global view of invariants and to refine events/variables/invariant specific to a component separately.

Taken separately, proof and simulation are powerful and useful techniques. But they are complementary and put together, as in ADVANCE technology, their power and usefulness is multiplied. Indeed, our Classical-B experience shows that models, even proved correct, are seldom suited for their purpose before being implemented and tested. Conversely, as everyone knows, non-proved non-trivial models are seldom correct. Testing models in realistic conditions, like we did it in the case study, allows validating their suitability; and proving suitable models allows verifying exhaustively their correctness. Thus ADVANCE provides the means to develop "by construction" valid and correct models. Compared to current practice this is a major technological breakthrough that will undoubtedly improve quality of systems and generate considerable savings as it is widely known that the most difficult and expensive errors to disclose and correct are system-level errors. ADVANCE technology and Classical-B define then an almost continuous and consistent formal development process, from system-level specification to software-level implementation.

Thanks to the ADVANCE project Alstom is able to develop formally its critical systems from system-level specification with Event-B to software-level implementation with Classical-B and improve their quality and its productivity.

5.3 Future exploitation of results of ADVANCE and the case study

We designed specifically the IXL-DC to meet the requirements of the Paris railway operator, RATP, regarding the full compliance of interlocking systems with system-level safety requirements. We shall then integrate the IXL-DC in our response to the tenders of RATP for the revamping of existing urban lines (4 and 11) and suburban lines (RER B) and for the construction of the new lines 15 to 18 of the "Grand Paris" project. These projects represent, as regarding the interlocking systems market alone, 20 to 30 million euros. Thus, in the short term the RATP market is our main target, but it is not the only one. If opportunities arise, we shall propose the IXL-DC to all railway operators that demand formal and exhaustive verification of interlocking systems, the New-York City Transport Authority for instance. In the medium term we should integrate the IXL-DC in all our interlocking systems, urban and mainline, and propose it to all railway operators.

The IXL-DC concept gave ideas of new and more innovative systems. Interlocking systems are built from a set of interlocking rules, also called "principles", specific to each country and even to each railway operator. For instance the Belgian, British, German and French interlocking principles are all different. Moreover, in France, the interlocking principles of RATP and SNCF (French mainlines railway operator) are different. This raises the "variability" problem of interlocking systems, i.e. modifying an interlocking system originally developed for a given operator for another operator with different interlocking principles leads to high costs of verification, validation and maintenance. Because the IXL-DC ensures compliance with safety requirements (almost) independently of the interlocking principles it is used in a project whose aim is to generate complete interlocking systems independently of interlocking principles.

In ADVANCE Deliverable D1.4 [4] we explain how we integrated ADVANCE technology in Alstom's system development process and how we shall exploit this technology. Engineers will be trained to ADVANCE methods and tools according to the needs of projects. The training material is available or is being developed by the ADVANCE partners.

Some of the tools developed or improved in ADVANCE will be used for activities not directly related with development of Event-B models. This is the case of ProB. We shall use this tool to check the compliance of actual configuration system data to formal configuration data rules. We shall also use ProB to discharge the proof obligations of Event-B and Classical-B models that can be discharged by examining, in reasonable time, all their possible instantiations.

To create the IXL-DC model we created and proved a mathematical theory of graphs. For the proof of that theory and of the IXL-DC model we created proof rules dealing with standard mathematical operators: (closure, union, UNION, dom, ran, card, $\in$, $\subseteq$, $\exists$, {x |P}, etc.). The graph theory shall be reused in other models because graphs are extensively used in railway models. The proof rules will be integrated in our proof rule data base and thus will be reused for the proof of other Event-B and Classical-B models. Reuse of these objects will save considerable efforts in the development of systems and software.

References

[1] Case Study in Railway Domain. ADVANCE project. Deliverable D1.1 – Workpackage 1. September 25th 2012.
[2] Proof of Concept Application in Railway Domain. ADVANCE project. Deliverable D1.2 – Workpackage 1. September 25th 2012.
[3] Intermediate Report on Application in Railway Domain. ADVANCE project. Deliverable D1.3 – Workpackage 1. October 21st 2013.
[4] ADVANCE Certification Strategy in Railway Domain. ADVANCE project. Deliverable D1.4 – Workpackage 1. November 30th 2014.
[5] ADVANCE project. Deliverable D5.3 – Workpackage 5. November 30th 2014.

[6] CENELEC Standard EN 50126: Railway applications — The specification and demonstration of Reliability Availability, Maintainability and Safety (RAMS); 1999.
[7] CENELEC Standard EN 50128: Railway applications — Communication, signalling and processing systems -Software for railway control and protection systems; October 2011.
[8] CENELEC Standard EN 50129: Railway applications — Communication, signalling and processing systems —Safety related electronic systems for signalling. February 2003.
[9] Constraints of the certification process. D1.1, OpenCoss: Open Platform for the Evolutionary Certification of Safety-critical systems, 28 March 2012. EC 7th Framework Programme.
[10] Performing hazard analysis on complex software and human-intensive systems. John Thomas, Nancy G. Leveson. Proceedings of the International System Safety Society Conference, Las Vegas, 2011.
[11] Engineering a Safer World. Systems Thinking Applied to Safety. Nancy G. Leveson. The MIT Press, January 2012.
[12] FP7 ADVANCE project, Proof of Concept Application in Railway Domain, D1.2.