Safety Analysis of a CBTC System: A Rigorous Approach with Event-B

Conference Paper · October 2017

DOI: 10.1007/978-3-319-68499-4_10

6 authors, including:

David Déharbe
ClearSy System Engineering
114 PUBLICATIONS **940** CITATIONS
SEE PROFILE

Julien Molinero Perez
ClearSy System Engineering
4 PUBLICATIONS **33** CITATIONS
SEE PROFILE

Denis Sabatier
ClearSy System Engineering
13 PUBLICATIONS **100** CITATIONS
SEE PROFILE

Safety Analysis of a CBTC System: A Rigorous Approach with Event-B

Mathieu Comptier[1], David Deharbe[1], Julien Molinero Perez[1], Louis Mussat[2], Thibaut Pierre[1], and Denis Sabatier[1]

[1] ClearSy System Engineering firstname.lastname@clearsy.com
[2] RATP firstname.lastname@ratp.fr

Abstract. This paper describes a safety analysis effort on RATP's communication-based train control (CBTC) system Octys. This CBTC is designed for multi-sourcing and brownfield deployment on an existing interlocking infrastructure. Octys is already in operation on several metro lines in Paris, and RATP plans its deployment on several other lines in the forthcoming years. Besides the size and complexity of the system, the main technical challenges of the analysis are to handle the existing interlocking functionalities without interfering with its design and to clearly identify the responsibilities of each subsystem supplier. The distinguishing aspect of this analysis is the emphasis put on intellectual rigor, this rigor being achieved by using formal proofs to structure arguments, then using the Atelier B tool to mechanically verify such proofs, encoded in the Event-B notation.
With this approach, we obtain a rigorous mathematical proof of the safety at system level—a level that is usually covered by informal reasoning and domain expert knowledge only. Such proof is thus feasible and it brings to light and precisely records the knowledge and know-how of the domain experts that have designed the system.

1 Introduction

Formal proof, instrumented with a formal method such as Event-B [1] and the accompanying software Atelier B, has been shown to be a powerful tool to perform rigorous safety analysis at the system level [6, 5].

Paris metro operator RATP has mandated ClearSy to perform a safety analysis of an existing CBTC system, named Octys. The aim is to prove that the system meets the safety goals: absence of collisions and derailments caused by uncontrolled switches, no over-speeding, and passenger safety. The Octys CBTC system is an assembly of subsystems, including for instance wayside and carborne computers, also linked to external subsystems such as the wayside interlocking. ClearSy's task is to mathematically assert the safety by proving the above safety goals, based on the detailed specifications of these subsystems. We bring out the reasoning ensuring these goals, first in an informal but completely rigorous way, then we turn these reasonings into Event-B models proven with the Atelier-B tool. The role assigned to each subsystem has to be based only on what we find

in the subsystems requirements, possibly with addenda or precisions if some requirements turn out to be ambiguous or incomplete.

The Octys CBTC system is particularly fit for this, being the standard system for the automation of existing RATP lines, carried out by a brownfield migration to Grade of Automation level 3 (GoA 3), i.e. train operation is mostly automatic, with a human pilot responsible for starting the train and taking over driving in case of emergencies. Octys is defined through a set of interoperability specifications allowing the use of subsystems from different independent suppliers. Every global function has been carefully split into precise roles for each subsystem, and the resulting requirements have gone through a considerable work, to ensure that each subsystem can be seamlessly purchased from any compliant vendor. This implies in particular very detailed interface specifications. Conversely, this carefully defined decomposition constitutes a system level design that has a paramount global impact, including on the safety. So it would be possible, and undesirable, that all subsystems match their Octys requirements, but that safety issues still remain at the system level. Such pitfalls are difficult to detect and solve, and, in case of failure, the responsibilities would likely be that of the system integrator, namely RATP. This motivates the system level mathematical proof presented here.

In this proof-oriented approach, we produce not only Event-B models and Atelier B proofs, but more importantly textual documents, at least for the sake of usability by users unfamiliar with Event-B. These documents identify the set of requirements from the original subsystem specifications used to perform the proof: because the proof was possible, we know that this set is at least sufficient. All the needed precisions, complements or disambiguations are also listed and explained. What if some requirements from the Octys specifications are not in this set, in particular if they are marked as safety critical? In such cases their role for the safety has to be carefully reviewed: either they are not needed, or something has been missed.

In this paper, we first present details about Octys. Then we describe the organization needed to perform such a proof. To give the reader an insight into how such a proof works, we expose some example mechanisms involved in the safety of Octys and how their rigorous proof is possible. We will also explain a key point: how it is possible to insert the existing interlocking in such a proof, without a detailed knowledge of its legacy design, considering its paramount role in the safety for such brownfield CBTCs. Finally, we discuss the benefits of this approach and the use of the results.

2 The CBTC System Octys

RATP (Paris Transport Authority) is undertaking a vast project to gradually upgrade their subway lines with driver. Accordingly, a *Communication Based Train Control* (CBTC) solution [3, 7] named Octys, for *Open Control of Train Interchangeable and Integrated System*, has been deployed since 2010. As a CBTC system, its main goals are to improve throughput and safety by ensuring contin-

uous train speed control, to participate in ensuring the safety of passenger transfers through the train and platform screen doors, to diminish the headway and to reduce wayside signaling requirements. Also, Octys relies on multi-sourcing and interchangeability. Indeed, the system is split in different sub-parts that are to be developed by different suppliers and interchangeable as any compliant subpart, whatever its supplier, shall fit seamlessly in the system. Octys has been deployed successively on Paris lines 3, 5 and 9; two other lines are scheduled to be equipped in the near future. Since service should not be disrupted during the migration, a key challenge is to maintain a good level of line availability.

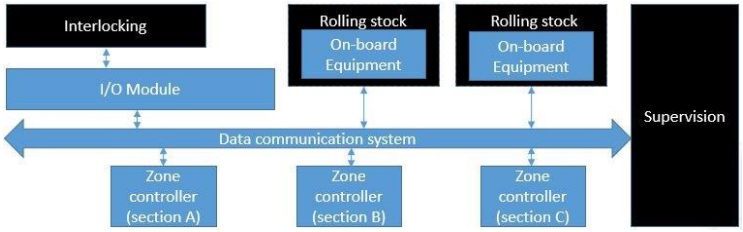

Fig. 1. Octys decomposition into subsystems. The blue blocks represent the wayside CBTC systems, the carborne CBTC systems and the data communication system between the wayside and carborne CBTC systems. The black boxes represent the existing sub-parts to be renewed to operate with the CBTC solution.

The CBTC has four different subsystems: the train controllers, the zone controllers, the data communication system and the I/O modules. The train controller is an on-board equipment that estimates the position of the train on the line, according to the cartography, signals from beacons installed along the track, and on-board odometric sensors. This calculated position is communicated to the zone controller subsystem. The second main function of the train controller is to continuously calculate and control the maximum speed authorized for the train depending on movement authority sent by the zone controller. The zone controller uses the train localizations and the interlocking system informations (track circuits, spot detectors, ...) to compute a track occupation mapping. This track occupation is then used to compute movement authority limits for each automatic train. These limits are calculated to avoid train-to-train collisions and derailments over uncontrolled switches. The I/O modules interface zone controllers with the interlocking for data such as track circuits occupation states or signal states. While the communication with the zone controllers is message-based, the interface with interlocking is analog. Finally, the data communication subsystem provides the communication infrastructure between the other equipments.

The specificity of the Octys CBTC is its adaptation to the RATP signaling system: in Octys, the interlocking remains a separated system with a legacy

design. Octys functions are tailored so that they fit with this existing design, introducing groups of automated trains in spaces already protected by the interlocking. Indeed, Octys mainly relies on the interlocking to guarantee that trains will not encounter unlocked switches, face-to-face collisions or side collisions over switches (in particular, the Octys CBTC does not move the switches, they remain under the sole control of the interlocking). This close interaction has an important impact on our system level proof as we want it to be independent from the legacy interlocking design; we had to carefully formulate the safety ensured by the interlocking before addition of the CBTC and check that all the modifications involved by the CBTC indeed preserve the system safety. Two examples are presented in section 4.

3 Methodology

3.1 Organization

This project involves three partners: the operator of the system, the safety analysis team, and a solution expert. The system operator is *RATP* and involves experts in both formal methods and railway systems. These experts also have access to the configuration of the existing lines where Octys is deployed and are able to answer clarification requests with respect to the system requirements. The analysis team is from *ClearSy* and initially started with three persons. It gradually built up to a group of five engineers (not all of them working full time on the project), with different technical background, but all with a strong knowledge of formal methods. The solution expert partner is *Siemens* and has a deep knowledge of Octys.

The main input for the analysis consists essentially of a dozen documents in PDF format, on the following subjects: functional and technical system specification describing the overall architecture, the top-level requirements and the main functionalities; specification of each subsystem with the specific hypotheses and requirements; interfaces and communication protocols between the different subsystems; interface between Octys and interlocking; system parameters; the system data base; and the rationale for some (but not all) design choices for the interface between Octys and interlocking. Each such source document is mainly textual, with illustrative diagrams and a few tables.

The partners have monthly meetings where the agenda consists mainly of presentations by the safety analysis team members. Such presentations typically revolve around a specific property: the corresponding Octys mechanism is introduced, illustrated with different scenarios and the draft of the proof mechanism is presented. This serves several purposes: to obtain clarifications and to verify that there is no misunderstanding with respect to the input documents, to validate argument hypotheses, to present proof mechanisms. Also, partners interact regularly by phone or by email, on an on-demand basis, mostly to provide to the safety analysis team the elements they need to conduct their work. Within the safety analysis team, informal discussions over technical issues occur routinely.

3.2 Approach

The goal is to produce a well-founded argument readable by anyone familiar with the main mechanisms of a CBTC, where safety appears as a logical consequence of a set of verified hypotheses. The absence of collision and of derailment derives from precise properties that hold for every possible event. This allows to conclude safety with a network topology that is neither known nor static. In a nutshell, we associate to every train a so called *train protection zone* where it is guaranteed to stay by its own braking forces, if nothing but the train changes on the track. If such zones are safe from collision and derailment, then we may conclude globally. An approach would be to study the evolution of the train protection zones caused by the interlocking and the CBTC functions. However only the CBTC functions are known and documented, whereas the sole hypothesis for interlocking is its safety when no CBTC is added. The key is to prove that each CBTC function leaves train protection zones safe, under the assumption that all the mechanisms of interlocking, that may occur concurrently, are safe.

Initially, some, but not all members of the safety analysis teams, had a strong background in CBTC systems. Of course none of them had previous knowledge of the specifics of Octys. Therefore the initial stage consisted in leveling the domain expertise across the team and in understanding the system. This was achieved by reading the source documents, and by producing so-called *exploratory scenarios*, i.e. simulating the system functionalities according to their understanding of the specification. Such scenarios are validated by the solution expert.

The safety analysis follows a hierarchical decomposition, guided by a top-level analysis of the different levels of protection zones and the impact of the different functions of Octys on these protections. The result is a collection of related safety arguments, each addressing a different target property. When expressing properties, our formal approach proves beneficiary as it demands an unambiguous and meaningful statement. A safety argument is a rigorous demonstration demanding a global understanding of the system. Such demonstration is based on hypotheses, which are justified with unsafe scenarios that would happen in case they do not hold. In some cases, hypotheses are considered as terminal, if close enough to a realistic truth (i.e. it is stated in the input document, is a physical property, or is submitted to validation). Otherwise, an additional demonstration is needed (see examples in section 4).

For each analyzed property, a specific Word document is produced, according to the following template: a *front matter* identifying the document, its author(s) and history; an *introduction* describing the property, the corresponding mechanism, its role in Octys, and a list of reference documents; a statement of the *target property*, with an exposition of the technical aspects of Octys related to this property; a compendium of all *hypotheses* needed to argue that the property stands; the *rigorous demonstration* of the target property under the given hypotheses; the *formalization in B* of this demonstration, given as a set of formal models that have been mechanically verified with the Atelier B tool; a lexicon of *frequently used notions* useful to simplify explanations.

Hypotheses. There are three kinds of hypotheses. Firstly, an hypothesis may be a pointer to a requirement, or set of requirements, found in the source documentation. This is the most common kind of hypotheses, and may come together with a request to change the wording of the requirement. Secondly an hypothesis may be an implicit assumption that needs to be made explicit. It must be submitted to the validation by an authority in the corresponding field. Thirdly, an hypothesis may be a new property about some mechanism of the CBTC. In that case, a proof of the property is necessary and a new document needs to be produced.

Demonstration. It is in textual form, and is sometimes illustrated with pictures. For clarity, entities may be represented by identifiers and conditions stated in mathematical form, yet the argument is presented in a such way that it can be read, followed and verified by an engineer without expertise in Event-B.

Formalization. The Event-B models formalizing the demonstration are included in the document, together with comments. Event-B is a text-based formalism to model systems by both specifying their properties, using classical logic, and describing their behavior, as event-based state machines [1]. Although Event-B is application domain-agnostic, its language includes simple mathematical entities useful for system modeling: integers, sets, relations, sequences to mention but a few. Event-B enforces that the user verify the behavior is consistent with the properties, by including a systematic generation of so-called proof obligations that the user must discharge, with the help of certified automatic and interactive theorem provers. The Event-B models that this project produces are usually of a small to moderate size and little effort is required from the user to discharge the proof obligations. The reason is that each model captures the essence of an argument establishing a specific property and nothing else, this argument has been established beforehand. In some cases, this formalization uncovers a corner case that has been omitted in the original argument. It usually requires little work to rectify the argument and the corresponding Event-B model.

How do we ensure that Event-B models do not miss a technical detail, rendering the proof irrelevant? Imagine for instance some kind of back-door functionality requested in a remote part of a document, that would bypass the proven mechanisms. There is obviously no other method to avoid this than to go through all the source documents. We do this exhaustive coverage in a traced way; for each document section we verify that it *does not contradict* what was modeled. Indeed, as we model only what is needed to ensure the target safety properties, functional and performance related mechanisms are not detailed. To give a rough example, we model that a train will stop before its movement authority limit but we do not model anything predictive about how it could accelerate. Nevertheless we do model that, when not braking in emergency, the train may accelerate at any time, so all functionalities linked to starting train movements (while not braking in emergency) in the source documents do not contradict the models. So we have to check this for all models and all the source documents—a demanding task, but the grouping of source documents by topics is of great help

here. If a requirement was given in the wrong source document regarding this topic splitting, it would not be applicable correctly anyway.

4 Illustrative examples

4.1 Track circuits backup example

As stated in section 2, railway interlocking systems use sensor devices, such as track circuits, to detect trains. For instance, interlocking maintains a switch locked as long as occupation of the track portion containing this switch holds, ensuring protection against derailment on an uncontrolled switch (a top-level safety property).

When a track circuit (TC) fails, it falls back to the occupied state. For interlocking, this means that this track portion is potentially unsafe and that trains should be prevented from entering it. Consequently, this fall-back behavior preserves safety, but at the expense of availability. To improve throughput, Octys has a function to back up TC occupancy, based on a logical tracking of trains done by the zone controller. When Octys TC backup is enabled, interlocking sees the track portion free if one of the sensor state and the output of this backup function is free.

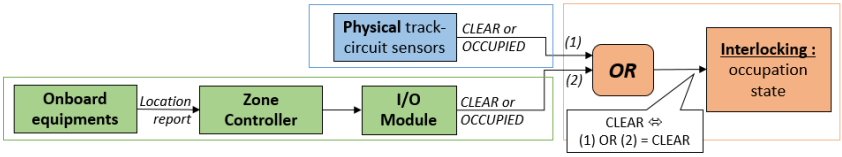

Fig. 2. Flow of information for the track circuit backup.

The backup modifies the timing between the detection of a train by a TC and the acquisition of the information by interlocking. The traditional path corresponds to the relay drop time (approximatively 200 ms). With backup, the path requires additional time due to communication and processing. Therefore, safety cannot be guaranteed without a careful analysis.

Without hypotheses on maximum train speed, minimal train length or minimal TC size, this new "occupation delay" may be so large that a train could have left the TC area when it is seen occupied by interlocking. In theory, this consideration exists already with the legacy system, but with a maximum delay of 200 ms, it is clearly not an issue in practice. This is no longer the case when the backup interfere, so a detailed analysis to prove that, when TC backup is enabled, interlocking still ensures protection against derailment has been performed. Essentially this analysis boils down to show whether the new delay is

small enough to ensure that interlocking will always follow train progression based on TC information. This provides us the following *target property* (see also figure 3).

> When a train circulates on an oriented track portion, covered by a set of TCs $N_0, N_1, \ldots, N_k$, there exists continuously a so-called "trailing track circuit" having the following properties:
> – interlocking sees it as occupied;
> – the tail of the train is downstream the area covered by this TC.

The existence of such trailing TC ensures that interlocking maintains locked the switches not yet crossed by the train and consequently protects the train progression, as it did originally with TC sensors (see fig. 2).

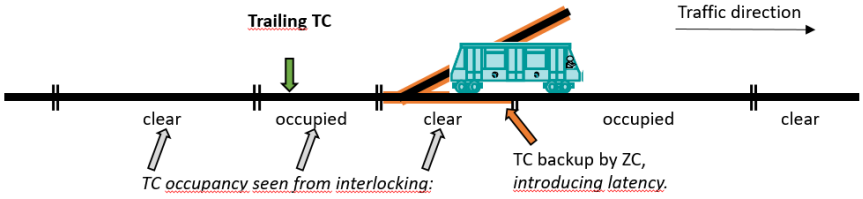

Fig. 3. Trailing TC.

The *proof* of this property uses induction and consists in showing that, assuming N_0 is a trailing TC at time t_0, then there exists a t_1 such that $t_0 \leq t_1$ and both N_0 and N_1 are trailing TCs. A simplified version of this argument follows. Let us introduce a few notations: N being a TC,

– $T_{\mathrm{i}}(N)$: the time the train enters the area covered by N;
– $T_{\mathrm{o}}(N)$: the time the rear of the train leaves the area covered by N;
– $T_{\mathrm{occ}}(N)$: the time interlocking starts seeing N occupied;
– $T_{\mathrm{fre}}(N)$: the time interlocking starts seeing N free again.

This notation being set, we claim that:

$$T_{\mathrm{occ}}(N_1) \leq T_{\mathrm{fre}}(N_0) \tag{1}$$

The argument to establish those properties is based on the following hypotheses:

H1 Once a train has occupied a TC N, this TC takes a minimum delay $T_{\min} \geq 0$ before it is freed upon the train leaving completely the area covered by this sensor: $T_{\mathrm{o}}(N)+T_{\min} \leq T_{\mathrm{fre}}(N)$. *Explanation:* This is essentially equivalent to say that a TC cannot become free while under a train. Note that, otherwise, the legacy system would not have been safe.

H2 After a train gets on a TC N, this TC will eventually be seen occupied by interlocking before a maximum delay $T_{\max}$: $T_{\text{occ}}(N) \leq T_{\text{i}}(N) + T_{\max}$. *Explanation:* This is the same as stating that a train cannot "jump over" a TC. Again, otherwise the legacy system would not have been safe.

By applying **H1** to N_0 and **H2** to N_1, and assuming TCs are contiguous (**H3**), the additional hypothesis $T_{\max} - T_{\min} \leq T_{\text{o}}(N_0) - T_{\text{i}}(N_1)$ establishes inequality 1. This condition is sufficient to establish the proof. Essentially, it imposes a constraint on the auto-crossing time, i.e. the time it takes a train to cover, at full speed, the distance corresponding to its own length. Under the given set of hypotheses **H1**-**H3**, this quantity has to be less or equal to the difference between $T_{\max}$ and $T_{\min}$. Assuming that trains respect speed limits (**H4**), this leads to the following hypothesis about the possible values for the system parameters to be able to deploy safely Octys' TC backup function (**H5**):

$$T_{\max} - T_{\min} \leq \frac{\text{MinLength}}{\text{MaxSpeed}}$$

Comments. The actual proof of the property takes into account other system parameters, namely the gap of shunt between consecutive TCs and the non-shunting dimensions at the ends of the trains. The existence and the value of $T_{\min}$ and $T_{\max}$ must be verified by another study which needs to consider other functionalities of the CBTC. We have verified that the values of the system parameters in Octys are such that hypothesis **H5** is indeed fulfilled, even though neither the hypothesis nor its argument are explicit in the source document. The analysis has been coded as an Event-B model and validated with Atelier B, guaranteeing the correctness of the reasoning presented with no possible contest.

Our analysis based on formal proofs aims to uncover such arguments, done during the design phase, and to explicit all necessary hypotheses. For the backup function, we exhibited an hypothesis that constrains the possible values of the system parameters and demonstrated that it is necessary to fulfill it to avoid sequences of events leading to the derailment of a train over uncontrolled switches.

4.2 Emergency cancellation / nominal crossing example

In railway interlocking, setting a route reserves a space for a train, where it can progress with the guarantee that it is protected against collisions and derailments on uncontrolled switches. When an emergency cancellation is requested, the route signal turns red so that no train can enter it. But the route signal also turns red as soon as a train occupies the first TC of that route. In Octys, signal status is an input to several functions of the ZC. For automated Octys trains, localization uncertainties difficult identifying the cause of the signal change by the ZC. If the ZC wrongly identifies the cause of the signal change, it may either force an unnecessary emergency brake of the train or authorize it to progress on an unprotected route, as explained below.

First, consider a localized automated train K that crosses a signal S that turns red because K occupies the first TC after S. The new aspect of S is

communicated to the zone controller (ZC). Also, K being localized, its position is calculated by the on-board calculator then transmitted to the ZC. Note that this position is tainted with two uncertainties: the measurement, transmitted as well and known by the ZC; and the flight time of the messages. For the ZC, K has been localized some seconds ago between two extreme positions. In this scenario, the last position of K received by the ZC does not allow it to conclude that the train has occupied the first TC. Assessing the situation, the ZC cannot exclude the possibility that K is still approaching S, which could have turned red due to a cancellation. This uncertainty occurs nominally, and commanding the train to brake here would impair seriously availability. So the CBTC shall avoid stopping a train on a red signal if there is a doubt that this red signal may be caused by the progression of K.

Second, consider again a train K approaching a signal S, but now an emergency cancellation occurs, causing S to turn red. As in the previous scenario, the new aspect of S is communicated to the ZC, and due to the uncertainty in the localization information, the ZC may see the head position of K downstream S. Since the ZC cannot exclude the possibility that K has crossed the signal, it cannot command to brake immediately. If nothing is done though, K may cross S after the moment when the emergency destruction was executed. K would risk either derailing on a moving switch or colliding against another train engaged in a conflicting route set later.

The solution to this uncertainty is to delay braking after the signal turned red. Nevertheless, this delay shall imperatively end soon enough to ensure that K will be stopped before the route cancellation delay expires. As far as formalism is concerned, the proof consists in exhibiting the reason why K is safe, either if it is stopped soon enough, or if it crosses the signal before the deadline.

Synthesis. On the one hand, a supplier wishing to implement the ZC subsystem should obey rigorously the safety demand and command the train to brake before it is too late, on the other hand, it should also optimize the functionality for its system, by providing the widest possible window for the train to cross the signal. In order to meet both criteria, an unambiguous description of the last moment at which the ZC should command the braking is needed. Conveniently, one corollary result of the formal argument we developed for this case is the valuation of the maximum delay before a ZC commands the braking.

5 Discussion and lessons learnt

As presented before, for each target property we analyze the mechanisms and we find the "reason why it always holds" *before* writing Event-B models. However we discovered that the reasoning is far from being complete until the corresponding Event-B models are written and proved. It appears that it is very difficult, or nearly impossible, to obtain the expected rigor without formulating in some kind of mathematical language; Event-B and the Atelier B tool serve here as a test of rigor. Of course, we have to ensure that all aspects of the reasoning are

correctly captured in the B models: this is done by verifying that if we remove any required hypotheses according to the informal reasoning, the proof in the B model actually becomes impossible.

With this process we isolate a set of hypotheses sufficient to ensure each property; these hypotheses have to be requirements found in the input documents, sub-properties proved afterwards, or agreed precisions to be added. Our output documents detail these requirements and precisions: this is probably the most important benefit of this work, as it allows the identification of possible pitfalls and the improvement of the source documents. One interesting case is when some requirements marked as safety critical are not used in our reasoning. Then the topic must be carefully examined with the domain experts to find out what those requirements were meant for. This can be difficult as Octys is based on the design of existing CBTCs that were developed over a long period of time.

Anyway, the interaction maintained with the domain experts is of paramount importance. This is not an easy topic: the proof team in this effort arrives after the design, as a kind of independent assessment in an already multi-supplier context, and the proof directly deals with the know-how and the know-why of this design. Any pitfall or needed precision that we find will be correctly taken into account only if the involved domain experts have enough time to carefully check these findings and are convinced. Then this contribution is perceived as a benefit, not a burden.

The ideal solution would be that the proof team be present with the design team from the time of what we could call the "prospective design ready" phase, i.e. when the design exists but is not yet finalized. There the notions found by the proof team could be directly used by the design, with benefits in obtaining the system level safety as well as in optimizing the subsystem requirements, potentially leading to an easier development of subsystems. We think that the proof team shall remain separated from the design team, as proof as well as design are demanding tasks that deserve a full dedication and because the proof team shall remain neutral regarding the choice of design solutions. A frequent and trusted interaction between the teams, although not necessary, would make the approach even more efficient and beneficial.

In the case of a multi-supplier interoperability specification like Octys, the interaction between design and proof is more complicated: the system level design is stabilized and the subsystems' design is the responsibility of suppliers, either existing or future. In this context the subsystem properties used as assumptions in our proofs could become target proof goals for the design of these parts. RATP uses formal methods to assert the correctness of such subsystems' design, particularly at software level by verifying the software source code of the supplier. So the output proof assertions from the system level proof could be used as target properties for this software formal verification. This is not yet done but certainly forecasted.

6 Conclusion

We have presented an on-going effort to analyze the safety of a CBTC system designed for deployment on existing interlocking using a rigorous approach. This approach consists in expressing properties key to the system safety, both in natural language and mathematical notation, and in constructing formal proofs that these properties hold. During the construction of these proofs, all necessary hypotheses are identified, be they explicitly stated in the specification of the system, physical laws, or requirements appearing to be missing from the specification.

While this effort is yet unfinished, it is now clear that such system level proof is feasible for a system like the Octys CBTC, with the appropriate level of independence with the intricate but out of scope interlocking. The findings are discussed in an on-going basis and already provide their benefits. In addition, the output results are expected to be used as input properties for subsystem formal analysis performed by RATP [2].

We forecast that the future Octys instantiations will put into light the benefits of this proof, through the presence of well established, strong safety related reasonings and the absence of system level pitfalls or doubts. We believe that this work will stimulate the application of this kind of system level proof for industrial projects.

References

1. Abrial, J.R.: Modeling in Event-B: system and software engineering. Cambridge University Press (2010)
2. Bonvoisin, D.: 25 years of formal methods at RATP. From manual approach for proof of programs to instrumented demonstration of railway systems safety (2016)
3. Forioni, S.: An innovative approach and an adventure in rail safety. Computer Engineering Series, Wiley (2014)
4. Lecomte, T., Pinger, R., Romanovsky, A. (eds.): Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification - 1st Int'l Conf., RSSRail 2016, LNCS, vol. 9707. Springer (2016)
5. Sabatier, D.: Using formal proof and B method at system level for industrial projects. In: Lecomte et al. [4], pp. 20–31
6. Sabatier, D., Burdy, L., Requet, A., Guéry, J.: Formal proofs for the NYCT line 7 (Flushing) modernization project. In: Derrick, J., Fitzgerald, J.S., Gnesi, S., Khurshid, S., Leuschel, M., Reeves, S., Riccobene, E. (eds.) Abstract State Machines, Alloy, B, VDM, and Z - 3rd International Conf'., ABZ 2012, Pisa, Italy. LNCS, vol. 7316, pp. 369–372. Springer (2012)
7. Tremblin, C., Lesoille, P., Rezzoug, O.: Use of Formal Proof for CBTC (Octys). Computer Engineering Series, Wiley (2014)