Statistical Model Checking of a Moving Block Railway Signalling Scenario with Uppaal SMC

Experience and Outlook

Davide Basile[1,2], Maurice H. ter Beek[1(✉)], and Vincenzo Ciancia[1]

[1] ISTI–CNR, Pisa, Italy
[2] University of Florence, Florence, Italy
{basile,terbeek,ciancia}@isti.cnr.it

Abstract. We present an experience in modelling and statistical model checking a satellite-based moving block signalling scenario from the railway industry with Uppaal SMC. This demonstrates the usability and applicability of Uppaal SMC in the railway domain. We also propose a promising direction for future work, in which we envision spatio-temporal analysis with Uppaal SMC.

1 Introduction

The railway sector is well known for its robust safety requirements. In fact, the CENELEC EN 50128 standard [31] for the development of software for railway control and protection systems specifically mentions formal methods as highly recommended practices for software systems to be certified at Safety Integrity Levels (SIL) 3 and 4. Indeed, formal methods and tools are widely applied to railway systems [17,37,34,36,18,35,12,48,6].

Also the Shift2Rail Joint Undertaking [53] (http://shift2rail.org), the first European rail initiative for focussed research and innovation under Horizon 2020 to increase competitiveness of the European rail industry through the development of safe and reliable technological advances to complete the single European railway area, considers formal methods to be fundamental for its ambitious aim: "double the capacity of the European rail system and increase its reliability and service quality by 50%, all while halving life-cycle costs."

In particular, specific calls were issued concerning the application of formal methods in supporting the transition to the next generation of ERTMS/ETCS signalling systems, which will include satellite-based train positioning, moving block distancing and automatic driving. The European Railway Traffic Management System (ERTMS) is a set of international standards for the interoperability, performance, reliability, and safety of modern European rail transport [30]. It relies on the European Train Control System (ETCS), an automatic train protection system that continuously supervises the train, ensuring to not exceed the safety speed and distance. The current standards distinguish four levels (0–3) of operation of ETCS signalling systems, depending largely on the role of trackside

equipment and on the way information is transmitted to and from trains. Full-fledged Level 3 systems are under development, but have not yet been deployed.

In this paper, we report our trial experience in modelling and statistical model checking a Level 3 moving block signalling scenario with Uppaal SMC. This task was performed in the context of two projects with ample participation from the railway industry, one of which funded under the above mentioned H2020 Shift2Rail initiative. Our experience shows that Uppaal SMC facilitates an easy transformation of semi-formal UML models into formal models (viz. stochastic timed automata). Furthermore, subsequent analyses with Uppaal SMC turned out to be very useful in discussions with our industrial partners, demonstrating the usability and applicability of Uppaal SMC in the railway industry. As an outlook for the future, we envision spatio-temporal analysis based on Uppaal SMC by exploiting its powerful continuous time analysis capabilities, especially its ODE modelling tools, and its statistical model-checking algorithms in combination with the typically discrete models produced by spatial analysis.

The remainder of the paper is organised as follows. Section 2 introduces the industrial context: next generation ERTMS/ETCS satellite-based moving block railway signalling systems. Section 3 first describes the specific case study, a railway signalling scenario with satellite-based positioning and moving block distancing, after which it presents Uppaal models of the case study and some preliminary results of applying the statistical model checking features of Uppaal SMC to the case study. Section 4 discusses a promising direction for future work, spatio-temporal analysis with Uppaal SMC, after which the contribution of this paper is briefly discussed in Section 5.

2 Industrial Context: Railway Signalling Systems

The ERTMS/ETCS signalling systems currently deployed on railways throughout Europe concern at most Level 2. These are characterised by the need for trackside equipment (such as track circuits or axle counters) only for exact train position detection and train integrity supervision, whereas communication of the movement authority (MA), speed information and route data to/from the train is achieved by continuous data transmission via GSM-R or GPRS with a Radio Block Centre (RBC). Moreover, an onboard unit (OBU) continuously monitors the transferred data and the train's maximum permissible speed by determining its position in between the Eurobalises (transponders on the rails of a railway) used as reference points via sensors (axle transducers, accelerometer and radar).

However, the current Level 2 signalling systems are still based on fixed blocks (sections of the railway track between fixed points), which start and end at signals, with their lengths designed to allow trains to operate as frequently as necessary (i.e., ranging from many kilometres for secondary tracks to a few hundred metres for busy commuter lines). The block sizes are determined based on parameters like the line's speed limit, the train's speed, the train's braking characteristics, drivers' sighting and reaction times, etc. But the faster trains are allowed to run, the longer the braking distance and the longer the blocks need

to be, thus decreasing the line's capacity. This is because the railway sector's stringent safety requirements impose the length of fixed blocks to be based on the worst-case braking distance, regardless of the actual speed of the train.

The next generation Level 3 signalling systems no longer rely on trackside equipment for train position detection and train integrity supervision, but an onboard odometry system is responsible for monitoring the train's position and autonomously computing its current speed. The OBU frequently sends the train's position to a RBC which, in turn, sends each train a MA, computed by exploiting its knowledge of the position of the rear end of the train ahead. For this to work, the precise absolute location, speed and direction of each train needs to be known. These can be determined by a combination of sensors: active and passive markers along the track, as well as trainborne speedometers.

The resulting moving block signalling systems allow trains in succession to close up, since a safe zone around the moving trains can be computed, thus considerably reducing headways between trains, in principle to the braking distance (cf. Fig. 1). This allows for more trains to run on existing railway tracks, in response to the ever-increasing need to boost the volume of passenger and freight rail transport and the cost and impracticability of constructing new tracks.

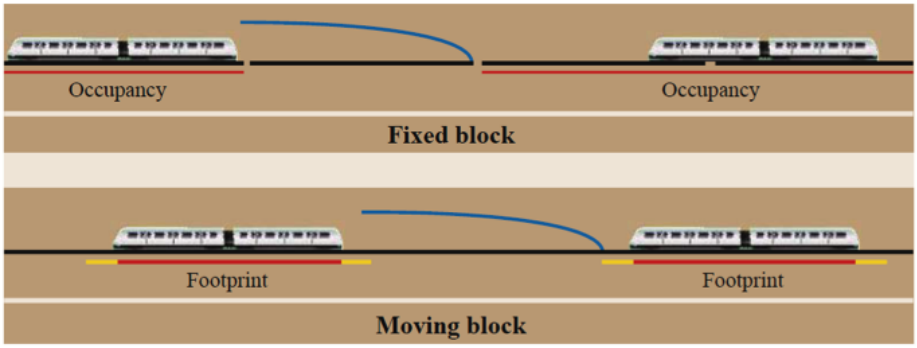


Fig. 1. Safe braking distance between trains for fixed block and moving block signalling

The envisioned future switch from Level 2 to Level 3 signalling systems would not only optimise the exploitation of railway lines due to the adoption of moving block signalling, but the removal of trackside equipment would result in lower capital and maintenance costs. However, compared with other transport sectors, the railway sector is notoriously cautious about the adoption of technological innovations. This is typically attributed to its well-known robust safety requirements. Therefore, one of the current challenges in the railway sector is to make moving block signalling systems as effective and precise as possible, including GNSS-based satellite positioning and leveraging on an integrated solution for signal outages (think, e.g., of tunnels) and the problem of multipaths [51].

Requirements analysis followed by safety, hazard and performance analyses of moving block signalling scenarios by means of formal methods and tools are a few of the topics addressed in the EU's H2020 Shift2Rail project ASTRail (S**A**tellite-based **S**ignalling and Automation Sys**T**ems on **Rail**ways along with Formal Method and Moving Block Validation) (http://www.astrail.eu) [11]. Moreover, one of the aims of the Tuscany region's project SISTER (SIgnaling & Sensing TEchnologies in Railway applications) (http://stlab.dinfo.unifi.it/sister-project) is to apply innovative signalling solutions in the context of Light Rail Transit infrastructures. In the next section, we describe a concrete case study that has been considered in these projects for a trial application of formal modelling and analysis in order to assess the usability and applicability of formal methods and tools in the railway domain. This assessment is one of the goals of the ASTRail project and currently considered to be an important issue for the successful uptake of formal methods and tools in the railway industry (cf., e.g., [11,48,47,6]).

3 Case Study: Moving Block Signalling Scenario

An important task of a work package of the ASTRail project is devoted to formal verification of a moving block signalling system, based on a model in Real-Time UML (RT UML) [29,52] provided by the project's industrial partners. This model takes as parameters the probability of failures of the different devices involved in the system (e.g. GNSS receivers), to be instantiated with data provided by the vendors of such devices. Hence, our task mainly consisted of translating the semi-formal UML model to a formal one, amenable to formal verification. We chose to use Uppaal SMC for this task, since it allows for both real-time and probabilistic aspects, which both occur in RT UML models. Moreover, the formalism is similar to UML state machine diagrams, which eased understanding by our partners. The visualisation of message sequence charts helped in this aspect. We now outline the moving block signalling scenario, after which we present the Uppaal models, and some hints on how we obtained them from the RT UML models, in Section 3.1 and the results of statistical model checking in Section 3.2.

The main components of the Level 3 moving block signalling system that we consider are a trackside RBC and a train's OBU and localisation unit (LU). The OBU measures the train's current speed and verifies the train's integrity, while the LU uses a GNSS-based positioning system to determine the train's location. The RBC is continuously in communication with the train's onboard units to receive data regarding the train's position and its integrity from the train, and to send speed restrictions, route configurations and MAs to the train. The RBC computes the latter by communicating with neighbouring RBCs and by exploiting its knowledge of the positions of switches and other trains (head and tail position) by communicating with a Route Management System (RMS). The model abstracts from the RMS and from the communication between neighbouring RBCs. Instead, it considers the train to communicate with one RBC, based on a seamless handover when the train moves from one RBC supervision area to the adjacent as regulated according to its Functional Interface Specification [54].

A (preliminary) hazard analysis was performed by our industrial partners in order to evaluate the safety level of a moving block signalling system. To this aim, hazards derived from the moving block signalling system in operation, such as GNSS-related errors, communication failures and faulty states, were identified and analysed, after which their risk level was assessed. This safety assessment concerned establishing the probability of the occurrence of a hazard and the severity of its consequences as well as risk qualifying according to the appropriate CENELEC EN 50126 standards concerning Reliability, Availability, Maintainability and Safety (RAMS) [32,33]. This results in a hazard log. We derived a number of safety properties from this hazard log, to be verified on the formal model. One of these will be analysed in Section 3.2, where we will provide some details from the log. The full hazard log is omitted for reasons of space.

3.1 Modelling

Uppaal SMC [28] is a variant of Uppaal [14], a well-known toolbox for the verification of real-time systems modelled by (extended) timed automata, which was introduced specifically for modelling and analysing cyber-physical systems. Uppaal SMC models are stochastic timed automata, in which non-determinism is replaced with probabilistic choices and time delays with probability distributions (uniform for bounded time and exponential for unbounded time). As usual, these automata may communicate via broadcast channels and shared variables.

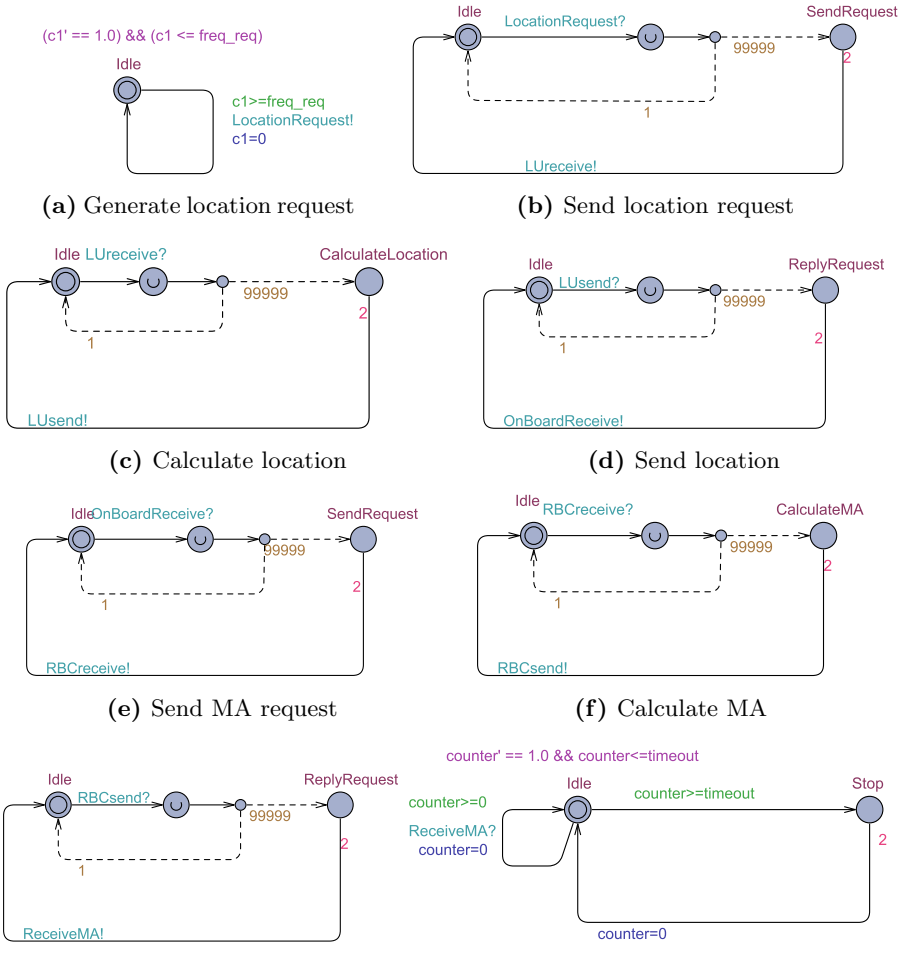
We transformed the RT UML state machine diagrams as provided by our industrial partners into stochastic timed automata. This model transformation was rather straightforward, except for a few issues that mostly concerned the precise meaning of (time-related) modelling choices and which had to be cleared during meetings with our partners. Before presenting our formal Uppaal models, we list the modelling choices and assumptions that we have made for the specific RT UML model at hand. The model itself is omitted for reasons of confidentiality.

Each parallel region of the RT UML model is translated into a separate automaton. (Pseudo) states and (probabilistic) transitions are in a one-to-one correspondence, except for the addition of urgent states[3] that are used to split communication actions from probabilistic choices. The failure probabilities are currently set to a placeholder value of 10^{-5}, but we recall that these are to be refined based on input from our project partners. Guards and triggers are modelled as input and output broadcast channels, respectively, which implies that we assume the different system components (i.e. the OBU, LU and RBC) to communicate in a synchronous manner. This means that a message is discarded in case the receiver is not ready (i.e. in the right state) to receive it. The main intuition behind this choice is based on the fact that in this way, a fresh MA sent by the RBC to the OBU will supersede any older MA in case the latter was not yet received.

[3] In timed automata, an urgent state (indicated with a '$\cup$' inside the state) is a state with no delay, which allows to reduce the number of clocks in a model, and thus the analysis' complexity.

Time-related aspects are rendered as follows. Timed events `RTat` of stereotype `<<RTevent>>`, typically used to trigger transitions based on the event's timing information, are modelled as invariant conditions and clock guards, which force transitions to be executed when the precise moment in time has been reached. Probabilistic delayed events `RTduration` of stereotype `<<RTdelay>>`, typically used to add durations to actions/transitions, are modelled as probabilistic delays: when an action/transition is enabled, the time at which it is fired is probabilistically distributed. As for failure probabilities and rates of probabilistic distributions, also these will be further refined based on input from our project partners.

We now briefly describe the eight stochastic automata, depicted in Fig. 2, that together make up our UPPAAL model of the moving block scenario.



(a) Generate location request **(b)** Send location request

(c) Calculate location **(d)** Send location

(e) Send MA request **(f)** Calculate MA

(g) Send MA **(h)** Control MA freshness

Fig. 2. Complete UPPAAL model of moving block signalling scenario

The automata are screenshots of the automata designed with UPPAAL SMC. In the figures, initial states are indicated by a double circle (e.g. `Idle` in Fig. 2h) and (non-urgent) states have a name (e.g. `Stop`) in purple and an invariant (possibly with clock rates) in pink (e.g. `counter'==1.0 && counter<=timeout`); transitions have guards in green (e.g. `counter>=timeout`), synchronisation actions in cyan (e.g. `ReceiveMA?`, while τ actions are omitted) and updates in blue (e.g. `counter=0`). Exponential distributions rates for states with unbounded delay are depicted in red (e.g. `2` in Fig. 2f), while the values of discrete probabilistic choices (whose transitions are displayed with dashed lines) are depicted in bone (e.g. `99999`, i.e. probability $\frac{99999}{1+99999}$). The probabilities are input parameters (originating from the industrial partners), whose values are the result of preliminary evaluations. More fine-tuning of these parameters is awaiting further input from our industrial project partners.

In brief, the automata operate as follows. At fixed intervals of time, given by the parameter `freq_req`, the train's OBU generates a location request (Fig. 2a), which is sent to the train's LU (Fig. 2b). Upon receiving this location request, the LU calculates the train's location (Fig. 2c) and responds to the OBU (Fig. 2d). Upon receiving the train's location, the OBU sends it—together with a requests for a MA—to the RBC (Fig. 2e). Once the RBC has received the location of the train and the request for a MA, it calculates the MA (Fig. 2f) and responds (by sending the MA) to the train's OBU (Fig. 2g). When the OBU has received a MA, it activates a timer to control the freshness of the MA and it activates the emergency stop whenever the current MA has become outdated, recorded by a parameter `timeout` (Fig. 2h). Our industrial partners set `freq_req` = 5 seconds and `timeout = 3*freq_req` = 15 seconds. Moreover, we set the initial value of the clock named `c1` to `freq_req` and the one named `counter` to 5, i.e. the transition of the automaton depicted in Fig. 2a is enabled initially.

The current model is rather simple, but we anticipate further complexity once we consider more than one train and more than one RBC.

3.2 Analyses

Statistical Model Checking (SMC) [45,44,1] is concerned with running a sufficient number of (probabilistic) simulations of a system model to obtain statistical evidence (with a predefined level of statistical confidence) of the quantitative properties to be checked. SMC offers advantages over exhaustive (probabilistic) model checking. Most importantly: SMC scales better, since there is no need to generate and possibly explore the full state space of the model under scrutiny, thus avoiding the combinatorial state-space explosion problem typical of model checking, and the required simulations can trivially be distributed and run in parallel. This comes at a price: contrary to (probabilistic) model checking, exact results (with 100% confidence) are out of the question. Another advantage is its uptake in industry: compared to model checking, SMC is very simple to implement, understand and use, and it requires no specific modelling effort other than an operational system model that can be simulated and checked against (state-based) quantitative properties.

In addition to standard model-checking queries concerned with reachability and deadlock-freeness, UPPAAL SMC allows to check (quantitative) properties over simulation runs of an UPPAAL SMC model (i.e. a network of stochastic timed automata). These properties must be expressed in the Weighted Metric Temporal Logic (WMTL) [19] defined by the grammar

$$\phi \ ::= \ p \ \mid \ \neg\phi \mid \ \phi_1 \wedge \phi_2 \ \mid \ X\phi \ \mid \ \phi_1 \, U_{x \leq t} \, \phi_2$$

where p is a state predicate, x is a clock and $t \in \mathbb{N}$. Negation and conjunction are the classical logic operators that yield the full propositional logic. X is the usual neXt operator from temporal logic: $X\phi$ states that in the *next* state of a simulation run, the formula ϕ is satisfied. U is a time-bounded Until operator: $\phi_1 \, U_{x \leq t} \, \phi_2$ is satisfied if the formula ϕ_1 holds on a simulation run *until* the formula ϕ_2 is satisfied, and this must happen before the clock x exceeds time t. As usual, it is possible to derive (time-bounded) *eventually* and *always* operators. Let *true* denote $\phi \vee \neg\phi$. Then $\Diamond_{x \leq t} \, \phi = \mathit{true} \, U_{x \leq t} \, \phi$ and $\Box_{x \leq t} \, \phi = \neg \Diamond_{x \leq t} \, \neg\phi$.

Let $\mathbb{P}_M(\Diamond_{x \leq t} \, \phi)$ denote the probability that a random simulation run of a model M satisfies ϕ. UPPAAL SMC supports the evaluation of the following three types of queries over a model M:

Probability estimation $\mathbb{P}_M(\Diamond_{x \leq t} \, p)$?
Hypothesis testing $\mathbb{P}_M(\Diamond_{x \leq t} \, p) \geq P$? $(P \in [0, 1])$
Probability comparison $\mathbb{P}_M(\Diamond_{x_1 \leq t_1} \, p_1) \geq \mathbb{P}_M(\Diamond_{x_2 \leq t_2} \, p_2)$?

Additionally, UPPAAL SMC supports the evaluation of expected values of min or max of an expression that evaluates to a clock or an integer value:

Average min $E[\mathit{bound}; N](\mathit{min} : \mathit{expr})$
Average max $E[\mathit{bound}; N](\mathit{max} : \mathit{expr})$

where *bound*, for $n \in \mathbb{N}^+$, is either (1) an implicit time bound specified by $\leq n$, (2) an explicit bound by cost specified by $x \leq n$, where x is a specific clock, or (3) a bound on the number of discrete steps specified by $\# \leq n$. Furthermore, N is the number of runs, and *expr* is the expression to evaluate.

Evaluating safety Several of the hazards reported in the hazard log provided by our industrial partners report as causes communication failures. Exemplary requirements of such hazards read "Communications between RBC and OBU must be safe and continuously supervised, if the connection is lost an alarm must be triggered" and "OBU device must be SIL 4 device. Once OBU receives the alarm [...] it must immediately send an alarm to RBC", to be mitigated by "In case of communication loss enter in safe state mode". Each hazard also has an associated so-called Safety Related Application Condition, such as "If train position cannot be received within the maximum time limit, the OBU shall generate an alarm and must transit to degraded mode" and "If Train Integrity cannot be confirmed within the maximum time limit, the train shall be stopped". The recurring aspects in these hazards (viz. communications and safe state) have been modelled, so we can formalise such requirements.

To acquire familiarity with our UPPAAL model, we first checked the property:

$$A\Diamond(\texttt{ReplyMA.ReplyRequest} \parallel \texttt{Controlling.Stop})$$

This CTL formula states that it must always (A) be the case that eventually ($\Diamond$) either (choice operator $\parallel$) a MA is received, i.e. state `ReplyRequest` is reached (cf. Fig. 2g), or the train enters a safe state (i.e. state `Stop` in Fig. 2h). We verified this formula on our model with UPPAAL SMC, which reported its satisfaction. Hence, we know that one of the two aforementioned events happens at some point of any simulation run. However, the formula does not express whether this happens infinitely often. Indeed, compared to full CTL, UPPAAL does not allow nesting of path formulae. Nevertheless, given that our model is cyclic (i.e. it eventually returns to the initial state), in this particular case if the property holds then it will indeed hold infinitely often. We remark that the only reason why a MA may not be received is due to repeated communication failures.

In the remainder of this section, the probability of reaching a safe state and the freshness of a MA are measured using SMC. In particular, we used the academic version 4.1.19 (rev. 5649) of UPPAAL SMC with the following set-up for the statistical parameters (for all evaluated properties): lower and upper probabilistic deviation ($-\delta, +\delta$): 0.001; probability of false negative and false positive (α, β): 0.005; probability uncertainty (ϵ): 5.0^{-5}.

Evaluating the probability of reaching a safe state The above property in CTL does not express the time by which the MA must be received, nor the probability of entering a safe state. These aspects are taken into account by the next formula:

$$\mathbb{P}_M(\Diamond_{\leq(\texttt{timeout}-1)}\, \texttt{Controlling.Stop})$$

where M is the composition of the automata in Fig. 2.

UPPAAL SMC estimates the probability that the train will be in the safe state `Stop` before a `timeout` actually occurs to be in the interval [0,9.99994e-005], with confidence 0.995 and obtained from 59912 runs. This low probability is clearly a result that confirmed our expectations and thus pleased our industrial partners. Note that it is not possible to reach the fail-safe state `Controlling.Stop` before clock `counter` has exceeded `timeout`; hence, the more accurate the statistical parameters the closer to zero this probability will be.

Consequently, we decided to evaluate the following, slightly different, formula:

$$\mathbb{P}_M(\Diamond_{\leq(\texttt{timeout})}\, \texttt{Controlling.Stop})$$

In this case, the clock does actually reach the `timeout` value and the automaton Controlling in Fig. 2h thus switches to the safe state `Stop`. However, UPPAAL SMC estimates also this probability to be in the interval [0,9.99994e-005], with confidence 0.995 and obtained from 59912 runs. Hence, the probability for this to happen is invariantly low, which again pleased our industrial partners. Both these evaluations took around 5 minutes.

Evaluating the freshness of the MA It is also important to check the 'freshness' of the MA messages. Basically, the older this message is, the less reliable it is considered to be. According to the case study's requirements, the OBU attempts for three times to compute the train's location and receive the MA. In our model, the first attempt takes place at time 0, after which it tries again each 5 seconds until a timeout occurs at time 15. It is thus of interest to check which of the three attempts has a higher probability of success. In UPPAAL SMC, this can be verified by means of the following formula:

$$E[\leq \texttt{timeout}; 10000](max : \texttt{Controlling.counter})$$

which computes, in the interval of time of `timeout` (i.e. 15 seconds), the average of the maximum value of the clock named `counter`, using 10,000 runs. UPPAAL SMC estimates this average to be in the confidence interval 5.73866 ± 0.0327581, in just over 3 hours. As can be concluded from Fig. 2, the clock `counter` is reset each time a new MA is received. Hence, its average value is the average time in which a new MA is received. The cumulative density distribution plot in Fig. 3 provides evidence that the MA messages have a higher probability of being received between the first and the second attempt.

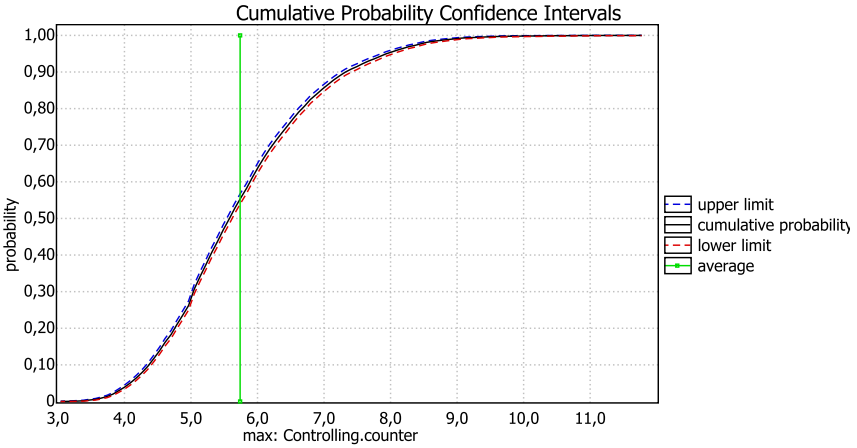


Fig. 3. Analysis of the freshness of the MA

Finally, we observe a trade-off between (1) the freshness of the MA and (2) the probability of reaching a fail-safe state. Indeed, by augmenting `timeout` an improvement in (2) and a deterioration of (1) is obtained (thus potentially leading to safety issues). Recall that the actual value instantiation for the waiting time (i.e. `freq_req` $= 5$ seconds) and the timeout (i.e. `timeout = 3*freq_req` $=$ 15 seconds) were provided by our industrial partners.

The analyses described in this section turned out to be very useful in discussions with our industrial partners concerning the usability and applicability of Uppaal SMC in the railway domain, as well as to fine-tune the practitioners' semi-formal RT UML model.

4 Future Work: Adding a Spatial Dimension?

An important aspect—currently abstracted away from—in the Uppaal model of the moving block scenario, presented in Section 3.1, concerns using *spatial information* in the case study. An example of such information is the location of trains (their coordinates in a map), calculated by the localisation unit, and used by the RBC to compute the MA messages.

By making the spatial locations of trains explicit, it becomes possible to check whether or not the system satisfies properties of interest of the form "*where* does property ϕ hold?", where property ϕ could be, e.g., "the train is allowed in the current location". Moreover, spatial relationships between different properties could then be checked: "does ϕ hold *near to* where ψ holds?"or "are the locations where ϕ holds *surrounded by* locations where ψ holds?". If, for instance, property ϕ expresses the presence of a single train in a specific area and ψ expresses the absence of trains in a specific area, then such formulae could be used to check whether it is true that for each train (travelling at a specific speed) there are no other trains around it (given a specific diameter of distance). Such formulae can be useful, for instance, for guaranteeing a safety distance between trains during normal operation conditions, and for computing the MA messages.

Properties that combine spatial and temporal reasoning (so-called *spatio-temporal* properties) are particularly subtle. Consider, for instance, the difference between being *near to an entity that will eventually satisfy* ϕ, meaning that the location where we are *now* is close to a location where ϕ will possibly become true in the future, and being *eventually near to an entity that satisfies* ϕ, denoting that we may move, at some future time, to a location close to one that at the same time satisfies ϕ. The complexity of such requirements and their analysis has given rise to the recent research line of *spatio-temporal model checking*, which leverages classical formal methods—in particular, modal logics and model checking—to the modelling and analysis of systems distributed in physical space.

Spatio-temporal model checking is a form of model checking that, besides atomic properties, makes spatial information explicit by providing spatial connectives such as $\texttt{near}\,\phi$, interpreted on *points*, which is true at all points that have an edge in common[4] with a point satisfying ϕ. More complex operators can be defined by employing notions of reachability, such as the 'surrounded' operator, or by using spatial metrics (e.g. distance or measure) or by referring to regions of spaces, besides points. Most of these developments have their roots in the field of *topological spatial logics* (cf. [2] for a comprehensive reference). We refer the reader to [22], and the citations therein, for further details.

[4] We assume that the considered graphs are symmetric, although this is not generally needed in spatial model checking, especially not in the research line of [22].

Although one might think that spatial model checking could be reduced to temporal model checking, this is not the case, due to several differences. For instance, in spatial model checking it is natural to also deal with regions, i.e. contiguous sets of points, which are not usually of interest in temporal model checking. Another point is that 'past' and 'future' modalities are equally relevant in spatial model checking, whereas in temporal model checking it is frequent to deal only with 'future' modalities. Also, the so-called state-space explosion problem is not an issue in current applications of *purely spatial* model checking, since models are explicitly specified state-by-state. However, this does not entail that spatial model checking is inherently limited to small models. Spatial models are often constrained by several factors that limit their complexity, e.g. graphs are typically Euclidean or even just regular grids. A particular case is that of multi-dimensional images (e.g. medical images), where implementations of spatial model checking benefit from very important optimisations that allow several millions of spatial points to be analysed in just seconds.

The state-of-the-art in the computational usage of spatial logics comprises applications of *machine learning*, where the logical operators cater for a machine-readable (and machine-learnable) language with a precise spatial interpretation, and model-checking applications, where the logical language is meant to be easily understandable and hand-written by humans, as for temporal model checking. In the machine-learning research line, some authors (cf., e.g., [42,5]) proposed to employ logical formulae that represent spatio-temporal behaviour as the evolution of a quad-tree representation of a partitioned image. Such formulae may accurately represent complex behaviour; however, these are not meant for human-intelligible specification, but rather tailored to machine learning. In contrast, the topological approach to spatio-temporal model checking, to which one of the authors of the present paper has contributed (cf. [21,22,23]) is devoted to the study of topological operators (like 'near', 'reachable', 'surrounded'), and to the definition of efficient model-checking algorithms, catering for a simple, albeit expressive logical language that may be used to specify and verify requirements of systems whose behaviour depends on the spatial distribution of components.

Technically, in the approach of [21], a spatio-temporal model over a set of atomic propositions P consists of a graph $G = (N, E)$ with nodes N and edges E, a Kripke structure $K = (S, R)$ with states S and accessibility relation R, and a valuation of atomic propositions $v : (N \times S) \to 2^P$. Consequently, the valuation of a formula ϕ assigns a truth value to each pair n, s consisting of a point $n \in N$ and a state $s \in S$, with the intuitive interpretation that ϕ is true at point n in state s. Spatial and temporal formulae can be freely nested, and the computational complexity of the model-checking algorithm is linear in the product of the size of the formula, the number of the temporal states, and the number of points.

Recently, the topological approach was applied in the field of signal analysis [50] and in the context of *smart cities* and *smart transport*, in particular to *statistical spatio-temporal model checking* of bike-sharing systems [24] and to the analysis of networks of *smart buses* [20]). Topological languages can express complex properties of space, as witnessed by their application to the identification of regions of interest in medical imaging (in particular, brain tumours) [15,16].

The tool `topochecker`[5] is an experimental—but rather feature-complete—model checker capable of analysing branching spatio-temporal models using combinations of the classical temporal operators of CTL [26], the spatial operators of the *Spatial Logic of Closure Spaces* SLCS [23], and the region operators of [22].

The most important aspects of the current paper are indeed related to *continuous* time; this permits the modeller to free herself from discretisation of a temporal domain which is continuous by nature. In the near future, we aim at exploring the same kind of approach to specify spatio-temporal requirements of railway signalling systems. The challenge in pursuing this objective is the combination of *discrete* and *continuous* features that arises from the continuous time model we presented, and the typically discrete models produced by spatial analysis. In this domain, space is usually approximated either as a graph or as a *patch model*, which can be defined as a particular kind of graph model where nodes and arcs are arranged according to a regular grid, denoting adjacency of rectangular region ('patches') that represent a discretisation of physical space.

Using the spatio-temporal model checker `topochecker` and the Uppaal toolset separately to analyse space and time is not feasible, since in spatio-temporal analysis the spatial and temporal aspects are intertwined. Our current investigation is therefore focussed on integration of the discrete and continuous world in Uppaal, in order to exploit its powerful continuous time analysis capabilities, especially its ODE modelling tools, and its statistical model-checking algorithms. Our basic idea is to define and implement a fully automated program, meant to receive as inputs a statistical spatio-temporal logic formula ϕ and an Uppaal temporal model exhibiting spatial behaviour (e.g. the model we present in this paper, enriched with information about the position and speed of a train at each state, on a railroad map). The program is intended to produce as output a new model, enriched with Uppaal functions, a set of *observers*—namely, additional parallel processes that do not affect the functionality of the existing ones—and an Uppaal formula ϕ' predicating on the observers. Statistical model checking of ϕ' is intended to yield the requested result, namely the probability or truth value of ϕ (cf. Fig. 4).

Although we are mostly reporting on work in progress, we can already comment on two possible strategies to address the above challenge. In one approach, the primitives of a spatial-logical language can be encoded as Uppaal functions and the spatial structure can be represented as a graph directly, using variables of the model checker and a function to identify the neighbourhood relation between points. In this way, spatial properties are used in Uppaal formulae directly, as if they were atomic properties of temporal states. Such an approach has the advantage of simplicity. However, it requires the reimplementation of a spatial model-checking algorithm (akin to the one used in `topochecker`) in Uppaal. Moreover, efficiency of such a procedure, and the computational feasibility of this approach with large spatial structures, remains to be demonstrated. Furthermore, this approach is intrinsically limited, as it would be impossible to nest temporal formulae inside spatial connectives.

[5] cf. https://github.com/vincenzoml/topochecker and http://topochecker.isti.cnr.it.

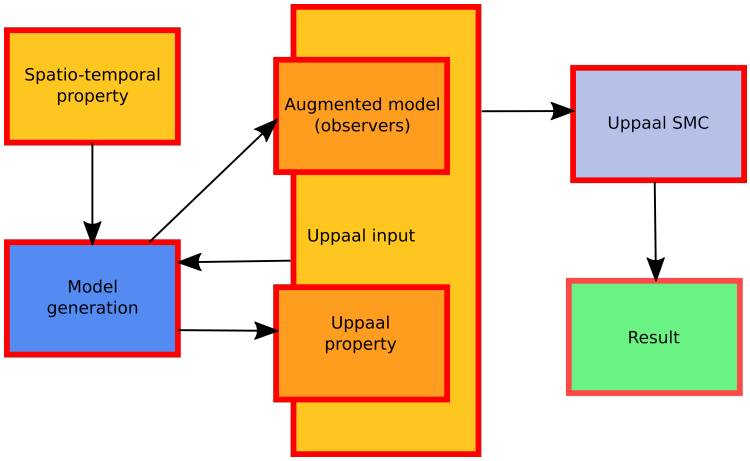


Fig. 4. Envisaged system architecture for spatio-temporal analysis with UPPAAL

For the above reasons, an apparently more interesting possibility is to encode space as an UPPAAL process, acting as a primary observer, so that spatial properties (e.g. reachability in space) can be checked by the algorithms of UPPAAL. Continuous clock variables can be used to represent movement in space, with each clock corresponding to a spatial dimension, and ODEs can be tied to spatio-temporal features of UPPAAL processes (position, speed, acceleration) in order to realise complex spatio-temporal analyses. However, this approach requires more work on the side of designing a suitable spatial language, and on defining appropriate observers that permit to represent nested spatio-temporal formulae that need to be encoded in the logical language used by UPPAAL.

We intend to test both approaches in the near future. Possible application scenarios are, e.g., those of *emergency egress* for train control systems. Consider, e.g., the situation in which, by some exceptional event, a train needs to be blocked and passengers ought to be rescued by a wheeled vehicle. This may, or may not, be possible, due to the situation of roads in the emergency situation, and centralised control is expected to instruct the train to stop in an appropriate place. A spatio-temporal property of interest is therefore to check whether the probability that the train gets blocked in a place where it cannot be reached by the rescue vehicle is low. Simulation can be used to model the spatio-temporal evolution of the emergency scenario. Such simulation must then be linked to the UPPAAL model we presented in order to use statistical spatio-temporal model checking to verify the properties of interest in the model.

5 Discussion

We presented a trial experience with modelling and statistical model checking a moving block signalling scenario from the railway domain with UPPAAL SMC and a future outlook towards spatio-temporal analysis with UPPAAL SMC.

First, we described the modelling and analysis of a concrete case study from the railway domain with Uppaal SMC. This trial application was performed with practitioners in industrial projects to assess the usability and applicability of Uppaal SMC in the railway sector. In particular, the modelling and analysis experience served to fine-tune their semi-formal model. This has set the stage for a more detailed model, with failure rates and probabilistic delays obtained from other project partners, which is still to be further formalised and analysed.

According to Uppaal SMC's case study page[6], most of the work on applying Uppaal SMC concerns the modelling and analysis of communication protocols and scheduling problems. There is another simplistic case study from the railway domain, but—rather than a next generation signalling system based on satellite-based positioning and moving block distancing—it considers a more classical railway control problem: "n trains wanting to cross a single track bridge which may be accessed by just one train at a time". Basically, the so-called *train gate case study* extends the original Uppaal model of [14] with arrival rates. Plain Uppaal was used in [41] to model and analyse the MA messages issued by the RBC in an ERTMS/ETCS Level 2 scenario of two trains. In the formal methods community at large, we witness a growing number of attempts at modelling stochastic or hybrid models of advanced ERTMS/ETCS Level 3 scenarios, generally applying 'real', i.e. not statistical, model checking [38,39,49,40,3,27,46,4].

We used our previous experiences with applying statistical model checking, and Uppaal SMC in particular, to case studies belonging to the transport domain [24,10,13]. Notably, in [10] we studied the reliability of systems of rail road switch heaters and their energy consumption, with the aim of comparing and tuning different policies of energy consumption [8]. This case study featured continuous physical aspects (heating equations modelled as an ODE) as well as discrete aspects (a communication protocol between the central control unit and the heating devices) and stochastic aspects (failure of a component and weather profiles). Stochastic hybrid automata and Uppaal SMC proved suitable to model and analyse this particular case study. Previously, in [7], we modelled the system through Stochastic Activity Networks and analysed it with the Möbius tool [25]. Starting from these two different formalisations, we then presented a technique to refine an automata-based discrete representation into a stochastic Petri net model to preserve safety through refinement in [9].

Finally, in this paper we also reported on a preliminary investigation of what we consider to be a promising new direction for the application of formal methods and analysis in the railway domain, and in the transport sector in general. Namely, we envision spatio-temporal analysis with Uppaal SMC by exploiting its powerful continuous time analysis capabilities, especially its ODE modelling tools, and its statistical model-checking algorithms in combination with the, typically discrete, models produced by spatial analysis, and the related *spatio-temporal model checking* algorithms. To this aim, we may take inspiration from [43], which proposes a methodology to discover potential safety hazards of transport systems already in the design phase, by verifying that the control software prevents collisions and fulfills certain spatio-temporal properties.

[6] http://people.cs.aau.dk/~adavid/smc/cases.html

Acknowledgements This work was partially funded by the Tuscany Region project SISTER and by the ASTRail project, which received funding from the Shift2Rail Joint Undertaking under the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 777561. The content of this paper reflects only the authors' view, and the Shift2Rail Joint Undertaking is not responsible for any use that may be made of the included information.

We thank our colleagues in the Formal Methods and Tools research group at ISTI-CNR, and the partners in these projects, for discussions on the models analysed in this paper.

References

1. Agha, G., Palmskog, K.: A Survey of Statistical Model Checking. ACM Transactions on Modeling and Computer Simulation **28**(1), 6:1–6:39 (2018). https://doi.org/10.1145/3158668
2. Aiello, M., Pratt-Hartmann, I.E., van Benthem, J.F.A.K.: Handbook of Spatial Logics. Springer (2007). https://doi.org/10.1007/978-1-4020-5587-4
3. Arcaini, P., Ježek, P., Kofroň, J.: Modelling the Hybrid ERTMS/ETCS Level 3 Case Study in Spin. In: Butler, M., Raschke, A., Hoang, T.S., Reichl, K. (eds.) Proceedings of the 6th International Conference on Abstract State Machines, Alloy, B, TLA, VDM, and Z (ABZ'18). LNCS, vol. 10817, pp. 277–291. Springer (2018). https://doi.org/10.1007/978-3-319-91271-4_19
4. Bartholomeus, M., Luttik, B., Willemse, T.: Modelling and Analysing ERTMS Hybrid Level 3 with the mCRL2 toolset. In: Howar, F., Barnat, J. (eds.) Proceedings of the 23rd International Conference on Formal Methods for Industrial Critical Systems (FMICS'18). LNCS, vol. 11119. Springer (2018). https://doi.org/10.1007/978-3-030-00244-2
5. Bartocci, E., Gol, E.A., Haghighi, I., Belta, C.: A Formal Methods Approach to Pattern Recognition and Synthesis in Reaction Diffusion Networks. IEEE Transactions on Control of Network Systems **5**(1), 308–320 (2018). https://doi.org/10.1109/tcns.2016.2609138
6. Basile, D., ter Beek, M.H., Fantechi, A., Gnesi, S., Mazzanti, F., Piattino, A., Trentini, D., Ferrari, A.: On the Industrial Uptake of Formal Methods in the Railway Domain. In: Furia, C.A., Winter, K. (eds.) Proceedings of the 14th International Conference on Integrated Formal Methods (IFM'18). LNCS, vol. 11023. Springer (2018). https://doi.org/10.1007/978-3-319-98938-9_2
7. Basile, D., Chiaradonna, S., Di Giandomenico, F., Gnesi, S.: A stochastic model-based approach to analyse reliable energy-saving rail road switch heating systems. Journal of Rail Transport Planning & Management **6**(2), 163–181 (2016). https://doi.org/10.1016/j.jrtpm.2016.03.003
8. Basile, D., Di Giandomenico, F., Gnesi, S.: Tuning Energy Consumption Strategies in the Railway Domain: A Model-Based Approach. In: Margaria, T., Steffen, B. (eds.) Proceedings of the 7th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications (ISoLA'16). Lecture Notes in Computer Science, vol. 9953, pp. 315–330. Springer (2016). https://doi.org/10.1007/978-3-319-47169-3_23
9. Basile, D., Di Giandomenico, F., Gnesi, S.: A Refinement Approach to Analyse Critical Cyber-Physical Systems. In: Cerone, A., Roveri, M. (eds.) Software Engineering and Formal Methods — Revised Selected Papers of the SEFM 2017

Collocated Workshops: DataMod, FAACS, MSE, CoSim-CPS, and FOCLASA. Lecture Notes in Computer Science, vol. 10729, pp. 267–283. Springer (2017). https://doi.org/10.1007/978-3-319-74781-1_19
10. Basile, D., Di Giandomenico, F., Gnesi, S.: Statistical Model Checking of an Energy-Saving Cyber-Physical System in the Railway Domain. In: Proceedings of the 32nd Symposium on Applied Computing (SAC'17). pp. 1356–1363. ACM (2017). https://doi.org/10.1145/3019612.3019824
11. ter Beek, M.H., Fantechi, A., Ferrari, A., Gnesi, S., Scopigno, R.: Formal Methods for the Railway Sector. ERCIM News: Research and Innovation **112**, 44–45 (2018), https://ercim-news.ercim.eu/en112/r-i/formal-methods-for-the-railway-sector
12. ter Beek, M.H., Gnesi, S., Knapp, A.: Formal methods for transport systems. International Journal on Software Tools for Technology Transfer **20**(3) (2018). https://doi.org/10.1007/s10009-018-0487-4
13. ter Beek, M.H., Legay, A., Lluch Lafuente, A., Vandin, A.: A framework for quantitative modeling and analysis of highly (re)configurable systems. IEEE Transactions on Software Engineering (2018). https://doi.org/10.1109/TSE.2018.2853726
14. Behrmann, G., David, A., Larsen, K.G., Håkansson, J., Pettersson, P., Yi, W., Hendriks, M.: UPPAAL 4.0. In: Proceedings of the 3rd International Conference on the Quantitative Evaluation of SysTems (QEST'06). pp. 125–126. IEEE (2006). https://doi.org/10.1109/QEST.2006.59
15. Belmonte, G., Ciancia, V., Latella, D., Massink, M.: From Collective Adaptive Systems to Human Centric Computation and Back: Spatial Model Checking for Medical Imaging. In: ter Beek, M.H., Loreti, M. (eds.) Proceedings of the Workshop on FORmal methods for the quantitative Evaluation of Collective Adaptive SysTems (FORECAST'16). Electronic Proceedings in Theoretical Computer Science, vol. 217, pp. 81–92 (2016). https://doi.org/10.4204/EPTCS.217.10
16. Belmonte, G., Ciancia, V., Latella, D., Massink, M., Biondi, M., Otto, G.D., Nardone, V., Rubino, G., Vanzi, E., Buonamici, F.B.: A topological method for automatic segmentation of glioblastoma in MR FLAIR for radiotherapy. Magnetic Resonance Materials in Physics, Biology and Medicine **30**(Suppl 1), 437 (2017). https://doi.org/10.1007/s10334-017-0634-z
17. Bjørner, D.: New Results and Trends in Formal Techniques and Tools for the Development of Software for Transportation Systems — A Review. In: Tarnai, G., Schnieder, E. (eds.) Proceedings of the 4th Symposium on Formal Methods for Railway Operation and Control Systems (FORMS'03). L'Harmattan (2003)
18. Boulanger, J.L. (ed.): Formal Methods Applied to Industrial Complex Systems — Implementation of the B Method. John Wiley & Sons (2014). https://doi.org/10.1002/9781119002727
19. Bulychev, P., David, A., Larsen, K.G., Legay, A., Li, G., Poulsen, D.B.: Rewrite-Based Statistical Model Checking of WMTL. In: Qadeer, S., Tasiran, S. (eds.) Runtime Verification — Revised Selected Papers of the 3rd International Conference on Runtime Verification (RV'12). Lecture Notes in Computer Science, vol. 7687, pp. 260–275. Springer (2013). https://doi.org/10.1007/978-3-642-35632-2_25
20. Ciancia, V., Gilmore, S., Grilletti, G., Latella, D., Loreti, M., Massink, M.: Spatio-temporal model checking of vehicular movement in public transport systems. International Journal on Software Tools for Technology Transfer **20**(3) (2018). https://doi.org/10.1007/s10009-018-0483-8
21. Ciancia, V., Grilletti, G., Latella, D., Loreti, M., Massink, M.: An Experimental Spatio-Temporal Model Checker. In: Bianculli, D., Calinescu, R., Rumpe, B. (eds.) Software Engineering and Formal Methods — Revised Selected Papers of the SEFM 2015 Collocated Workshops: ATSE, HOFM, MoKMaSD, and

VERY*SCART. Lecture Notes in Computer Science, vol. 9509, pp. 297–311. Springer (2015). https://doi.org/10.1007/978-3-662-49224-6_24
22. Ciancia, V., Latella, D., Loreti, M., Massink, M.: Model Checking Spatial Logics for Closure Spaces. Logical Methods in Computer Science **12**(4), 1–51 (2016). https://doi.org/10.2168/LMCS-12(4:2)2016
23. Ciancia, V., Latella, D., Loreti, M., Massink, M.: Spatial Logic and Spatial Model Checking for Closure Spaces. In: Bernardo, M., De Nicola, R., Hillston, J. (eds.) Formal Methods for the Quantitative Evaluation of Collective Adaptive Systems — Advanced Lectures of the 16th International School on Formal Methods for the Design of Computer, Communication, and Software Systems (SFM'16), Lecture Notes in Computer Science, vol. 9700, pp. 156–201. Springer (2016). https://doi.org/10.1007/978-3-319-34096-8_6
24. Ciancia, V., Latella, D., Massink, M., Paškauskas, R., Vandin, A.: A Tool-Chain for Statistical Spatio-Temporal Model Checking of Bike Sharing Systems. In: Margaria, T., Steffen, B. (eds.) Proceedings of the 7th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation: Foundational Techniques (ISoLA'16). Lecture Notes in Computer Science, vol. 9952, pp. 657–673. Springer (2016). https://doi.org/10.1007/978-3-319-47166-2_46
25. Clark, G., Courtney, T., Daly, D., Deavours, D., Derisavi, S., Doyle, J.M., Sanders, W.H., Webster, P.: The Möbius Modeling Tool. In: Proceedings of the 9th International Workshop on Petri Nets and Performance Models (PNPM'01). pp. 241–250. IEEE (2001). https://doi.org/10.1109/PNPM.2001.953373
26. Clarke, E.M., Henzinger, T.A., Veith, H., Bloem, R. (eds.): Handbook of Model Checking. Springer (2018). https://doi.org/10.1007/978-3-319-10575-8
27. Cunha, A., Macedo, N.: Validating the Hybrid ERTMS/ETCS Level 3 Concept with Electrum. In: Butler, M., Raschke, A., Hoang, T.S., Reichl, K. (eds.) Proceedings of the 6th International Conference on Abstract State Machines, Alloy, B, TLA, VDM, and Z (ABZ'18). LNCS, vol. 10817, pp. 307–321. Springer (2018). https://doi.org/10.1007/978-3-319-91271-4_21
28. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Poulsen, D.B.: Uppaal SMC tutorial. International Journal on Software Tools for Technology Transfer **17**(4), 397–415 (2015). https://doi.org/10.1007/s10009-014-0361-y
29. Douglass, B.P.: Real-Time UML. In: Damm, W., Olderog, E.R. (eds.) Proceedings of the 7th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT'02). Lecture Notes in Computer Science, vol. 2469, pp. 53–70. Springer (2002). https://doi.org/10.1007/3-540-45739-9_4
30. EEIG ERTMS Users Group: ERTMS/ETCS RAMS Requirements Specification — Chapter 2 - RAM (30 September 1998), http://www.era.europa.eu/Document-Register/Documents/B1-02s1266-.pdf
31. European Committee for Electrotechnical Standardization: CENELEC EN 50128 — Railway applications – Communication, signalling and processing systems – Software for railway control and protection systems (1 June 2011), https://standards.globalspec.com/std/1678027/cenelec-en-50128
32. European Committee for Electrotechnical Standardization: CENELEC EN 50126-1 — Railway applications – The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS) – Part 1: Generic RAMS process (1 October 2017), https://standards.globalspec.com/std/10262901/cenelec-en-50126-1
33. European Committee for Electrotechnical Standardization: CENELEC EN 50126-2 — Railway applications – The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS) – Part 2: Systems approach to

safety (1 October 2017), https://standards.globalspec.com/std/10262978/cenelec-en-50126-2
34. Fantechi, A.: Twenty-Five Years of Formal Methods and Railways: What Next? In: Counsell, S., Núñez, M. (eds.) Software Engineering and Formal Methods — Revised Selected Papers of the SEFM 2013 Collocated Workshops: BEAT2, WS-FMDS, FM-RAIL-Bok, MoKMaSD, and OpenCert. Lecture Notes in Computer Science, vol. 8368, pp. 167–183. Springer (2013). https://doi.org/10.1007/978-3-319-05032-4_13
35. Fantechi, A., Ferrari, A., Gnesi, S.: Formal Methods and Safety Certification: Challenges in the Railways Domain. In: Margaria, T., Steffen, B. (eds.) Proceedings of the 7th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications (ISoLA'16). Lecture Notes in Computer Science, vol. 9953, pp. 261–265. Springer (2016). https://doi.org/10.1007/978-3-319-47169-3_18
36. Fantechi, A., Fokkink, W., Morzenti, A.: Some Trends in Formal Methods Applications to Railway Signaling. In: Gnesi, S., Margaria, T. (eds.) Formal Methods for Industrial Critical Systems: A Survey of Applications, chap. 4, pp. 61–84. John Wiley & Sons (2013). https://doi.org/10.1002/9781118459898.ch4
37. Flammini, F. (ed.): Railway Safety, Reliability, and Security: Technologies and Systems Engineering. IGI Global (2012). https://doi.org/10.4018/978-1-4666-1643-1
38. Fränzle, M., Hahn, E.M., Hermanns, H., Wolovick, N., Zhang, L.: Measurability and Safety Verification for Stochastic Hybrid Systems. In: Proceedings of the 14th International Conference on Hybrid Systems: Computation and Control (HSCC'11). pp. 43–52. ACM (2011). https://doi.org/10.1145/1967701.1967710
39. Ghazel, M.: Formalizing a subset of ERTMS/ETCS specifications for verification purposes. Transportation Research Part C: Emerging Technologies **42**, 60–75 (2014). https://doi.org/10.1016/j.trc.2014.02.002
40. Ghazel, M.: A Control Scheme for Automatic Level Crossings Under the ERTMS/ETCS Level 2/3 Operation. IEEE Transactions on Intelligent Transportation Systems **18**(10), 2667–2680 (2017). https://doi.org/10.1109/TITS.2017.2657695
41. Ghosh, S., Dasgupta, P., Mandal, C., Katiyar, A.: Formal verification of movement authorities in automatic train control systems. In: Proceedings of the 5th International Conference on Railway Engineering (ICRE'16). pp. 1–8. IET (2016). https://doi.org/10.1049/cp.2016.0511
42. Grosu, R., Smolka, S.A., Corradini, F., Wasilewska, A., Entcheva, E., Bartocci, E.: Learning and Detecting Emergent Behavior in Networks of Cardiac Myocytes. Communications of the ACM **52**(3), 97–105 (2009). https://doi.org/10.1145/1467247.1467271
43. Hordvik, S., Øseth, K., Svendsen, H.H., Blech, J.O., Herrmann, P.: Model-Based Engineering and Spatiotemporal Analysis of Transport Systems. In: Maciaszek, L.A., Filipe, J. (eds.) Evaluation of Novel Approaches to Software Engineering — Selected Papers of the 11th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE'16). Communications in Computer and Information Science, vol. 703, pp. 44–65. Springer (2016). https://doi.org/10.1007/978-3-319-56390-9_3
44. Larsen, K.G., Legay, A.: Statistical Model Checking — Past, Present, and Future. In: Margaria, T., Steffen, B. (eds.) Proceedings of the 6th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation: Specialized Techniques and Applications (ISoLA'14). Lecture Notes in Computer

Science, vol. 8802, pp. 135–142. Springer (2014). https://doi.org/10.1007/978-3-662-45231-8_10
45. Legay, A., Delahaye, B., Bensalem, S.: Statistical Model Checking: An Overview. In: Barringer, H., Falcone, Y., Finkbeiner, B., Havelund, K., Lee, I., Pace, G.J., Rosu, G., Sokolsky, O., Tillmann, N. (eds.) Proceedings of the 1st International Conference on Runtime Verification (RV'10). Lecture Notes in Computer Science, vol. 6418, pp. 122–135. Springer (2010). https://doi.org/10.1007/978-3-642-16612-9_11
46. Mammar, A., Frappier, M., Tueno Fotso, S.J., Laleau, R.: An Event-B Model of the Hybrid ERTMS/ETCS Level 3 Standard. In: Butler, M., Raschke, A., Hoang, T.S., Reichl, K. (eds.) Proceedings of the 6th International Conference on Abstract State Machines, Alloy, B, TLA, VDM, and Z (ABZ'18). LNCS, vol. 10817, pp. 353–366. Springer (2018). https://doi.org/10.1007/978-3-319-91271-4_24
47. Mazzanti, F., Ferrari, A.: Ten Diverse Formal Models for a CBTC Automatic Train Supervision System. In: Gallagher, J.P., van Glabbeek, R., Serwe, W. (eds.) Proceedings of the 3rd Workshop on Models for Formal Analysis of Real Systems and the 6th International Workshop on Verification and Program Transformation (MARS/VPT'18). Electronic Proceedings in Theoretical Computer Science, vol. 268, pp. 104–149 (2018). https://doi.org/10.4204/EPTCS.268.4
48. Mazzanti, F., Ferrari, A., Spagnolo, G.O.: Towards formal methods diversity in railways: an experience report with seven frameworks. International Journal on Software Tools for Technology Transfer **20**(3) (2018). https://doi.org/10.1007/s10009-018-0488-3
49. Nardone, R., Gentile, U., Benerecetti, M., Peron, A., Vittorini, V., Marrone, S., Mazzocca, N.: Modeling Railway Control Systems in Promela. In: Artho, C., Ölveczky, P.C. (eds.) Formal Techniques for Safety-Critical Systems — Revised Selected Papers of the 4th International Workshop on Formal Techniques for Safety-Critical Systems (FTSCS'15). Communications in Computer and Information Science, vol. 596, pp. 121–136. Springer (2016). https://doi.org/10.1007/978-3-319-29510-7_7
50. Nenzi, L., Bortolussi, L., Ciancia, V., Loreti, M., Massink, M.: Qualitative and Quantitative Monitoring of Spatio-Temporal Properties. In: Bartocci, E., Majumdar, R. (eds.) Proceedings of the 6th International Conference on Runtime Verification (RV'15). Lecture Notes in Computer Science, vol. 9333, pp. 21–37. Springer (2015). https://doi.org/10.1007/978-3-319-23820-3_2
51. Rispoli, F., Castorina, M., Neri, A., Filip, A., Di Mambro, G., Senesi, F.: Recent Progress in Application of GNSS and Advanced Communications for Railway Signaling. In: Proceedings of the 23rd International Conference Radioelektronika (RADIOELEKTRONIKA'13). pp. 13–22. IEEE (2013). https://doi.org/10.1109/RadioElek.2013.6530882
52. Selic, B.: The Real-Time UML Standard: Definition and Application. In: Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'02). pp. 770–772 (2002). https://doi.org/10.1109/DATE.2002.998385
53. Shift2Rail Joint Undertaking: Multi-Annual Action Plan (26 November 2015), http://ec.europa.eu/research/participants/data/ref/h2020/other/wp/jtis/h2020-maap-shift2rail_en.pdf
54. UNISIG: FIS for the RBC/RBC handover, version 3.1.0 (15 June 2016), http://www.era.europa.eu/Document-Register/Pages/set-2-FIS-for-the-RBC-RBC-handover.aspx