Analysis of Relay Interlocking Systems via SMT-based Model Checking of Switched Multi-Domain Kirchhoff Networks

Roberto Cavada*, Alessandro Cimatti*, Sergio Mover‡, Mirko Sessa*†, Giuseppe Cadavero§, Giuseppe Scaglione§

*Fondazione Bruno Kessler
Trento, IT
{cavada, cimatti, sessa}@fbk.eu

‡University of Colorado Boulder
Boulder, USA
sergio.mover@colorado.edu

†University of Trento
Trento, IT

§Rete Ferroviaria Italiana
Rome, IT
{g.cadavero, g.scaglione}@rfi.it

***Abstract*—Relay Interlocking Systems (RIS) are analog electromechanical networks traditionally applied in the safety-critical domain of railway signaling. RIS consist of networks of interconnected components such as power supplies, contacts, resistances, and electrically-controlled contacts (i.e. the relays). Due to cost and flexibility needs, RIS are progressively being replaced by equivalent computer-based systems. Unfortunately, RIS are often legacy systems, hard to understand at an abstract level, hence the valuable information they encoded in them is not available.**

In this paper, we propose a methodology and a tool chain to analyze and understand legacy RIS. A RIS is reduced to a Switched Multi-Domain Kirchhoff Network (SMDKN), which is in turn compiled into hybrid automata. SMT-based model checking supports various forms of formal analyses for SMDKN. The approach is based on the modeling of the RIS analog signals (i.e. currents and voltages) over continuous time, and their mapping in terms of railways control actions. Starting from the diagram representation, we overcome a key limitation of previous approaches based on purely Boolean models, i.e. the presence of spurious behaviors. The evaluation of the tool chain on a set of industrial-size railway RIS demonstrates practical scalability.

I. INTRODUCTION

Railway signaling systems guarantee the safe operation of train traffic. Trains run between points of the rail network, moving from section to section along exclusively allocated routes and crossing roads. Protection against catastrophic events, such as train-to-train and train-to-car collisions, is devoted to various devices such as semaphores, barriers at the level crossing, and train detection systems. These devices must be suitably controlled and coordinated by a logic that ensures the safety of operation even in case of multiple device faults.

Traditionally, the logic has been implemented by means of the Relay technology, in the form of networks of interconnected analog electro-mechanical components, such as power supplies, contacts, circuit breakers, and many forms of electrically-controlled contacts, also known as *relays*.

RIS are progressively being replaced by computer-based logics (CBL), that ensure greater flexibility and lower cost. The key question is how to ensure that the CBL is compliant with the (trusted) behavior of the relay-based interlocking being replaced. In some sense, the specification for the CBL is hidden in the relay circuit. Unfortunately, RIS are often old, legacy systems, hard to understand for software engineers at the level of abstraction required to specify the CBL. Thus, the valuable information they encode is not readily available.

Although relays may be thought of as Boolean components, that is just open or closed, this turns out to be a gross simplification. In order to operate (e.g. switching from open to closed), relays may require time, and go through transients required to fully excite the circuitry. Hence, a simple Boolean propagation is in fact a coarse abstraction of a sequence of intermediate states before stability. Furthermore, relays are subject to faults that may either delay or prevent the correct operation. Thus, relay networks are often designed in a redundant fashion in order to mitigate the effect of faults and to ensure safety (at the cost of liveness) in all conditions.

In this paper, we propose a methodology and a tool chain to analyze and understand legacy RIS, adopted in an ongoing research project of Rete Ferroviaria Italiana (RFI). At the surface, a graphical tool supports the component-based modeling of the RIS. The designer selects components from a palette of over 100 elements, and connects them according to the input description – typically, a printout of the electrical schematic. This step does not require any deep understanding of the nature of the circuit, and ensures that the semantic gap w.r.t. the legacy description is as limited as possible. The corresponding internal representation is reduced to a Switched Multi-Domain Kirchhoff Network (SMDKN), which has a semantic based on Differential Algebraic Equations (DAE). In turn, the SMDKN is compiled into a network of hybrid automata, based on the techniques proposed in [1]. Then, various forms of formal analysis are supported by means of SMT-based model checking. At its core, the approach is based on the modeling of the RIS analog signals (i.e. currents and voltages) over continuous time. The ability to analyze the circuit at the physical level supports a comprehensive understanding at the symbolic level in terms of railways control actions. This is done by defining suitable symbolic predicates in terms of the analog state: for example,

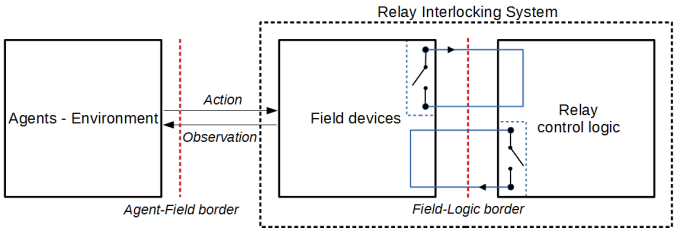

Fig. 1: Conceptual architecture of a RIS.

a green light to the train may correspond to a suitable current and voltage drop in the corresponding semaphore lamp.

The methodology is fully supported by an automated SMT-based verification tool chain. We evaluated the approach on a set of industrial-size railway RIS, with schematic having more than a thousand components and four-meter long plotter printouts. The results demonstrate practical scalability: we are able to prove (or disprove) conjectured properties, simulate scenarios, and construct fault-trees (FT) corresponding to undesirable events.

This approach was devised as a consequence of a previous unsatisfying modeling attempt we carried on basing our analysis on the traditional formal modeling at the Boolean level. Since relays are not instantaneous Boolean switches, substantial ingenuity from the modeler was required to bridge the gap with respect to the electrical semantics. This made the modeling task unmanageable in terms of conceptual hardness, and led to imprecise results (due to spurious behaviors) that we will report in the following sections. From a pragmatic perspective, the proposed approach provides invaluable support for the understanding of the legacy circuit (and ultimately the reverse-engineering of requirements for the CBL design).

The paper is structured as follows. In Section II we describe Relay Interlocking Systems. In Section III we overview SMDKN. In Section IV we describe the modeling approach. In Section V we present the analysis methods. In Sections VI and VII we present the tool chain and the experimental evaluation on a scalable industrial-size case study. In Sections VIII and IX we describe related work, draw some conclusions, and outline ongoing and future work.

II. RELAY INTERLOCKING SYSTEMS

A Relay Interlocking System (RIS) is an electromechanical system that conveys messages between the railway *agents* (e.g., trains, dispatchers, technicians). Fig. 1 shows the conceptual architecture of a RIS: the agents interacts with the *field devices* (e.g., semaphores, level crossing barriers, railroad switches) that are in turn controlled through the *relay control logic* (an interconnection of relays).

The agents interact with the field devices observing their state (e.g., if a semaphore light is on or off, the position of a barrier or of a railroad switch) and perform some actions (e.g., toggling an electrical contact, pushing a button) to change the current state of the RIS. The field devices are then connected to the relay control logic that reacts to the state change to

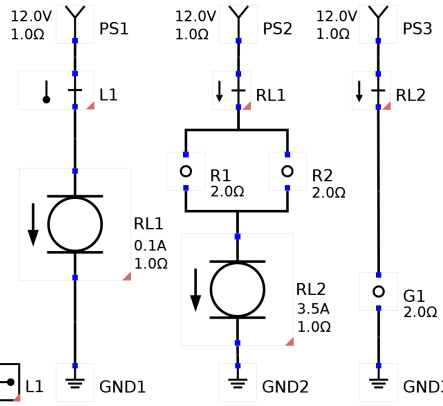

Fig. 2: Principle schemata of the RIS R2G1 that controls the semaphore lights for a RIS level crossing. The RIS is formed by 4 sub-circuits not connected electrically — from left to right: a lever handle, the lever sub-circuit, the sub-circuit that controls the red lights of the traffic semaphore, and a sub-circuit that controls the green light of the train semaphore.

implement the signaling system (e.g., lower the barrier of a level crossing when a train is approaching).

The RIS is implemented as a network of switching electromechanical components where relays are the main switching components. Relays are electrically-controlled analog switches that implement the relay logic. A relay contains a mechanical contact that can open or close a contact (e.g., a relay can open or close the circuit of a semaphore light turning it on or off). A relay controls its contact with a coil that is physically disconnected from the contact itself. The relay switches the contact when the current that flows in the coil falls within or exceeds a *current threshold*. The relay is in the *dropped state* when the coil's current is below the threshold and it is in the *drawn state* otherwise. When a component in a RIS switches to a different state, for example when an agent pushes a button, it induces different circuit contacts and hence a different behavior of the currents and voltages in the RIS. The changes in the currents and voltages can in turn change the state of the relays in the circuit (e.g., the change of the current on the relay coil switches the state of the relay). Thus, a single state change in the RIS may generate a sequence of subsequent state changes.

A *principle schemata* is the standard graphical representation[1] of the design of a RIS. Fig. 2 shows the principle schemata for the RIS that controls the semaphore lights for a level crossing (we will refer to this example as R2G1). In the RIS a lever handle (the component named L_1 in the lower left part of the diagram) controls the semaphore for the level crossing (the red lights R_1 and R_2) and the semaphore for the train track (the green light G_1).

Each connected set of components in the RIS represents a sub-circuit (i.e., sub-circuits are not connected electrically to each other). In Fig. 2 there are 4 sub-circuits — from left to

[1]We use the graphical representation defined in the Italian railway regulation UNIFER-CEI S-461 [2].

right, the sub-circuits are the lever handle L_1 (note that the lever handle is by itself a sub-circuit), the sub-circuit that is controlled by L_1, the sub-circuit that controls the red lights, and the sub-circuit that controls the green light.

The sub-circuits are not connected electrically (i.e., with a wire), but are "connected" with some other means (e.g., mechanically, as for a lever, or magnetically, as for a relay coil). A component on one sub-circuit (e.g., a relay coil) opens or closes its contacts (e.g., the relay contacts) that are on other (electrically disconnected) sub-circuits. The principle schemata separates the representation of the components (e.g., a relay coil) and their contacts (e.g., the relay contact). In Fig. 3 we show the symbols for a relay coil and its contacts. In a schemata, the components and their contacts are identified by name: the contacts for a relay coil named RL_1 will be also named RL_1. In a well formed schemata the same component name is used only for a component and its contacts (e.g., two relay coils cannot have the same name) and a contact must have a correspondent component (e.g., if a schemata has a contact named RL_1, it must also have a relay coil named RL_1). We say that there is a *logical connection* between a component and its contacts. The contact symbols in the diagrams further

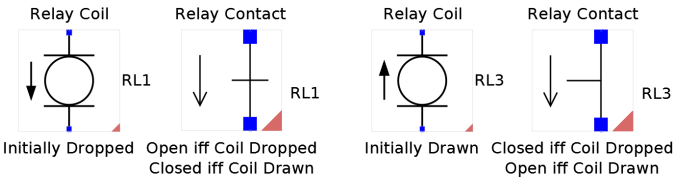

Fig. 3: Symbols of the relay coils and their contacts.

define when the contact should be open or closed. The two left-most components in Fig. 3 are the relay coil RL_1 and an "open" contact RL_1 (in this case, the "open" qualifier identifies a contact that is open by default). The downward arrow shown on the left of the "open" relay contact specifies what will be the state of the contact (i.e. open or closed) depending on the state of its relay coil. In Fig. 3, the contact RL_1 is open when the relay coil RL_1 is dropped and closed otherwise. Note that for the "closed" contact RL_3 of Fig. 3 the downward arrow specifies that the contact is closed when the relay coil RL_3 is dropped, and open otherwise.

The graphical representation of the components further defines the electrical terminals of the components with blue square boxes and the electrical connections among terminals with black solid lines. The orientation of a component (important to determine the physical position, such as if a lever in the left, center, or right position), is uniquely represented with a red triangle in the bottom right corner of the component. The graphical representation describes also the initial state of switching components like lever handles and relay coils. For relay coils (see Fig. 3) the initial state is determined by the upward or downward arrow at the left of the component, while for lever handles the initial state is the position (left, center, right) of the lever handle (e.g., in the schemata of Fig. 2, the lever handle L_1 is initially in the left position).

In the RIS R2G1 we further have other electrical components like power generators (PS_1, PS_2, and PS_3) that generate a current on the sub-circuit and "ground components" (GND_1, GND_2, and GND_3) that determine the ground for each sub-circuit. The lever "open" contact L_1 in the RIS R2G1 is further closed only if the lever handle L_1 is in the center position (see the position of the lever on the left of the L_1 contact in Fig. 2).

The RIS R2G1 implements a control logic that ensure that every time the green light is on (i.e. the train can travel through the track section with the level crossing), the red lights are also on (i.e. the cars have to stop at the level crossing). In the initial configuration of the RIS R2G1 both the red lights and the green lights are off. This is because the lever handle L_1 is in the left position, thus the lever contact L_1 is open, and hence no current flows in the sub-circuit and the coil RL_1 is dropped. Since the coil RL_1 is dropped, the contact RL_1 is open and no current flows through the red lights and the relay coil RL_2, which are respectively off and dropped. The contact RL_2 is further open and the green light is off. When an operator moves the lever handle L_1 to the center position she starts a sequence of state changes in the RIS.

1) The operator moves the lever handle L_1 to the center position. This change instantaneously closes the lever contact L_1, and the current starts flowing on the coil RL_1.
2) After a small amount of time (the "transient" time of the relay), the relay coil RL_1 switches from the dropped to the drawn state, and the relay contact RL_1 closes. At this point, some current flows on the red lights and on the relay coil RL_2. The red lights turn on.
3) After a small amount of time, the relay coil RL_2 switches to the drawn state and the relay contact RL_2 closes, powering the green light that turns on.

III. Switched Multi-Domain Kirchhoff Networks

Switched Multi-Domain Kirchhoff Networks (SMDKN) are a formalism that models a network of components connected according to the Kirchhoff conservation laws. SMDKN models systems where the components are from different domains (e.g., electrical, hydraulic, mechanical).

The components of a SMDKN are hybrid systems that change a set of discrete modes instantaneously, with a discrete transition, and the value of the physical variables (e.g., the current on a branch) continuously as a function of time. For each possible combination of the discrete modes of the components the SMDKN has a different continuous behavior. Technically, for each configuration the continuous behavior of the SMDKN is defined with a Differential Algebraic Equation derived from the behavior of each single component of the network and the Kirchhoff conservation laws.

IV. Modeling approach

A. Choosing the modeling abstraction level for relays

The physical behavior of a RIS is determined by the complex electromechanical phenomena of the relays. The "stationary" relay's states are the drawn and dropped states. However, the real behavior of a relay is more complex due to

inertial electromechanical phenomena: the transition between two stationary states is not instantaneous when the current on the relay's coil exceeds (or falls below) the threshold. Thus, we face the problem of modeling the relay's "transient states".

On the one hand a precise modeling of the "transient states" of the relays is challenging. First, such modeling requires complex differential equation; second, a RIS designer cannot reason precisely about the dynamic of the relay in the transient states. On the other hand, a purely "Boolean abstraction" approach that abstracts the physical quantities of the relay (e.g., the current on the coil) is also not adequate. Such abstraction does not permit reasoning about the physical quantities and the relative time between events.

We adopt an intermediate approach where we model the physical quantities of the system but we abstract the "transient state" of the relays. We model that after the relay's current crosses the threshold the change of state of the relay happens in a non-deterministic (but bounded) time interval. This time interval is a known design parameter of a relay. Our approach preserves the actual stationary physics of the system and enables automatic reasoning on the relative time distance between events, that are two key aspects for the designer. In our ongoing project we identified this abstraction level as the suitable trade-off between the designer's needs and the availability of precise and efficient model checking algorithms.

B. *Modeling* RIS *with* SMDKN

RIS are networks of components electrically connected by means of the Kirchhoff conservation laws. For this reason, we model RIS with SMDKN. The main advantages of the SMDKN modeling are: (i) **Preserve the RIS structure.** We model the RIS network as a SMDKN that has the same network structure (i.e. electrical connections on the components' terminals). Thus, RIS designer can easily model the RIS principle schemata as a SMDKN. (ii) **Compositional modeling.** SMDKN allow us to define the component behaviors independently. Our modeling effort is thus limited to create a library of components for the RIS domain. (iii) **SMDKN are an expressive and flexible modeling language**. SMDKN allow us to model the behavior of switching components as hybrid automata. With hybrid automata we can easily model the "abstraction level" described above. (iv) **Availability of formal analysis techniques**. There already exist efficient formal verification techniques SMDKN [3], [1] that we can apply off-the-shelf.

In the following, we describe in depth our modeling of the principle schemata as SMDKN, focusing on the components, their electrical connections, and the logical connections.

Components: we model a component in the RIS domain as a component in the SMDKN with a hybrid automaton. The hybrid automaton is standard [4]: it defines a finite set of discrete modes and continuous variables. In each discrete mode the automaton defines with a differential equation how the contiguous variables change in function of time, and with a conjunction of Boolean inequalities the invariant conditions. Transitions between discrete modes models the instantaneous state changes. Both RIS and SMDKN components have electrical terminals. We follow the standard approach in *acausal modeling* [5] to encode terminals with two variables, the flow and effort variables. In the electrical domain, the flow variable represents the current on the terminal, while the effort variable represent the potential on the terminal. Flow and effort variables will then be used to model the Kirchhoff conservation laws. The terminal implicitly has two continuous variables to represent flow and effort. Note that a component only exposes the effort and flow variables to the other components.

We describe in depth the modeling of a relay coil and of a faulty lamp. Both components are representative of the RIS library we developed that contains more than 100 components.

The model of the *delayed relay coil* shown in Fig. 4 follows the abstraction level described above where the transient states of the relay coil are modeled non-deterministically. The two modes *Dropped* and *Drawn* of the automaton represent two stable states where the coil has completely actuated its contacts. The two modes *Drawing* and *Dropping* encodes the transient states of the coil. The automaton uses a clock variable *clock* to encode the bounded and non-deterministic transition delays between the stable modes. In particular, the automaton transition from the *Dropped* to the *Drawn* mode only fires when the electrical current I through the coil continuously exceeds the current threshold I_{th} for a non-deterministic time within the specified time interval $[\Delta T_-, \Delta T_+]$. The same happens for the transition from the *Drawn* to the *Dropped* mode.

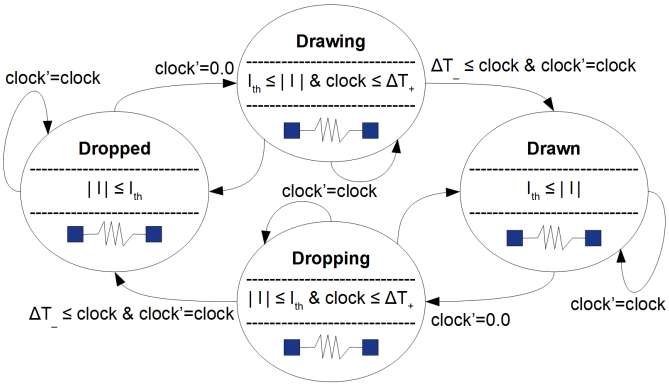

Fig. 4: Hybrid automaton of the delayed relay coil.

Fig. 5 shows the model of a *faulty lamp*, a lamp that can fail either creating a short-circuit or opening the circuit. The *Nominal* mode encodes the correct behavior of the lamp, which behaves as an ohmic load resistor. The automaton encodes the two fault conditions in the *FaultShort* and *FaultBlown* modes, where the lamp behaves respectively as a short-circuit and as an open circuit. The automaton can non-deterministically transition from the nominal mode to the two faulty modes. Since the lamp does not exhibit commutation delays, the hybrid automaton does not have continuous variables.

Physical connections: the semantics of the terminal connections follows the Kirchhoff's conservation laws. Given a set

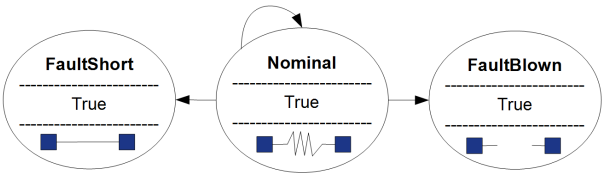

Fig. 5: Hybrid automaton of the faulty-lamp.

of connected terminals, all the effort variables of the terminals take the same value, and the sum of all the flow variables of the terminals equals zero. The SMDKN semantics already considers the Kirchhoff's law.

Logical connections: We model the logical connection among two components (e.g., a relay coil and one of its contacts) with additional synchronization constraints among the discrete modes of the hybrid automata of two components. For instance, for the relay coil RL_1 and the relay contact RL_1 of Fig. 3 the constraint encodes that the coil is in the *Dropped* mode if and only if the contact is in the *Open* mode, and in the *Closed* mode otherwise. Similarly, for the lever handle L_1 and lever contact L_1 of Fig. 2, we say that the handle is in the *Center* mode if and only if the contact is *Closed* mode.

Physical behavior of the running example: we present the relevant electrical behavior of the R2G1 system when lamps can fail either blown or short-circuited. The relay coil RL_2 of Fig. 2 senses the electrical current I_{PS_2} flowing through the parallel connection of the red lamps R_1 and R_2 in order to monitor their status. The current threshold of the coil RL_2 should be properly set to prevent inadvertent activation of the green lamp G_1 when the red lamps are either off or faulty. Tab. I shows the value of the current I_{PS_2} as a function of the 9 possible system modes resulting from the cross product of the 3 modes of the red lamps (see Fig. 5).

System mode	**Current $I_{\mathbf{PS}_2}$**
Both red lamps failed blown	0.0 Ampere
One red lamp failed blown, one red lamp nominal	3.0 Ampere
Both lamps nominal	4.0 Ampere
At least one red lamp failed short-circuited	6.0 Ampere

TABLE I: Values of the electrical current I_{PS_2} sensed by the relay coil RL_2 when the red lamps are power supplied by the closed relay contact RL_1.

To detect the simultaneous activation of the red lamps, the current threshold of the relay RL_2 must be set in the interval $]3.0A, 4.0A[$, for instance to $3.5A$. Notice that, in the system design of Fig. 2, the configurations "both lamps nominal" and "at least one red lamp failed short-circuited" are indistinguishable to the coil RL_2 because in both cases the current I_{PS_2} exceeds the coil threshold of $3.5A$. In the following section, we discuss the implication of this consideration on the overall system safety and we show how the proposed methodology supports the designer on this kind of quantitative reasoning.

V. Formal analysis

In a RIS, the agents determine their next action observing the state of the field devices. Thus, the agents observe a partial-state of the system because the internal state of the control logic is hidden from their point of view. Nevertheless, the correctness of the signaling protocol is implicitly dependent from the implementation of the relay logic.

In our methodology, we propose to analyze the system at two levels of detail: at the higher *railway level* we consider only high-level properties over the field devices (e.g., the lamp emits light, the barrier is closed), despite the technological details of the control logic; at the lower *physical level* we consider properties that investigate the internal technological aspects of the control logic and of its physics (e.g., two terminals must be short-circuited when a relay is in a specific mode). This layered approach reduces the total effort to specify properties: the properties at the railway level are independent from the implementation of the control logic and can be reused for multiple control logic implementations.

Properties specification: a property at the physical level predicates on low level aspects of the system such as physical quantities and operating modes of the components. Focusing on the electrical domain, we can predicate either on the voltage drop ΔV across a pair of terminals, or on the current I that flows through a terminal. A similar approach holds in the mechanical domain replacing *current* and *voltage* with *torque* and *angular velocity*. A property can further predicate on the operational modes of the components.

A railway property is automatically mapped onto a combination of physical properties, hiding its implementation details. For instance, consider the sentence *"the lamp* G_1 *emits light"*. Since a lamp is electrically equivalent to an ohmic load resistor, the property is equivalent to *"the lamp* G_1 *consumes electrical power"* that in turns is equivalent to the first-order logical formula $I_{\mathrm{G}_1} \neq 0.0 \wedge \Delta V_{\mathrm{G}_1} \neq 0.0$. Notice that in the context of physical reasoning it is necessary to predicate on *both* currents and voltage drops in order to distinguish the nominal behavior of the lamp from the faulty ones (i.e. those in which the lamp is power supplied, but does not emit light). In fact, a short-circuited lamp is traversed by a non-null current ($I_{\mathrm{G}_1} \neq 0.0$), but its voltage drop is zero ($\Delta V_{\mathrm{G}_1} = 0.0$); similarly, a blown lamp is traversed by a null current ($I_{\mathrm{G}_1} = 0.0$) even if its voltage drop is different from zero ($\Delta V_{\mathrm{G}_1} \neq 0.0$). In our specification settings, we could also refine the property exploiting detailed information available to the designer. Assuming to know the range of nominal currents absorbed by the lamp (e.g., from its data sheet), we could rewrite the predicate $I_{\mathrm{G}_1} \neq 0.0$ into a more precise one such as $1.5 \leq |I_{\mathrm{G}_1}| \leq 2.3$.

Analysis of the running example: in the following we demonstrate the need of the quantitative reasoning, which is enabled by our modeling approach, using the RIS R2G1 of Fig. 2. We further consider variants of the R2G1 model changing the fault model for the red lamps and the current threshold of the relay coil RL_2. The red lamps may either not fail, or the red lamps may blown (see the *FaultBlown* state in Fig. 5), or the red lamps can introduce a short circuit (see the *FaultShort* state in Fig. 5). The current threshold on the relay coil RL_2 may be either 2.5A, or 3.5A, or 4.5A. We

Nr.	R2G1 Variants		Verification results	
	Faults	RL_2 thresh.	RP	SP
1	None	2.5A	Hold	Hold
2	None	3.5A	Hold	Hold
3	None	4.5A	Doesn't hold	Hold
4	Blown	2.5A	Hold	Doesn't hold
5	Blown	3.5A	Hold	Hold
6	Blown	4.5A	Doesn't Hold	Hold
7	Short	2.5A	Hold	Doesn't hold
8	Short	3.5A	Hold	Doesn't hold
9	Short	4.5A	Hold	Doesn't hold

TABLE II: Verification results (property holds or does not hold) on variants of R2G1 introducing faults on the red lamps and changing the current threshold on the relay coil RL_2.

consider the reachability property RP := *"the green lamp* G_1 *can emit light"*, and the safety property SP := *"if the green lamp* G_1 *emits light, then both red lamps* R_1 *and* R_2 *emit light"*. We expect RP to hold for R2G1, witnessing an execution scenario where green lamp is on, and SP to hold to ensure the safety of the R2G1 system. The verification results are available in Tab. II.

When the current threshold of the relay coil RL_2 is over-dimensioned to 4.5A, the unexpected verification of the property RP proves that the green lamp cannot emit light because the relay contact RL_2 will never supply power to the lamp (rows 3, 6). Decreasing the threshold, RP always holds and this fact guarantees that the green lamp can turn on.

When the current threshold is under-dimensioned to 2.5A, the safety property SP is violated in the system variant with blown lamps (row 4). The counterexamples returned by the model checker provide execution scenarios able to reach the violation, but do not represent an exhaustive analysis. To determine all the minimal configurations of faults that lead to the violation, we perform formal safety assessment to compute fault-trees. For the system variant of row 4, the fault-tree of the safety property SP shows two possible fault configurations: when one red lamp fails blown, the other red lamp can still emit light absorbing 3.0A (see Tab. I) from the power supply PS_2. The 3.0A current exceeds the under-dimensioned threshold of 2.5A, thus the relay RL_2 inadvertently supplies power to the green lamp, violating the safety property. We fix this design flaw setting the coil threshold to 3.5A (row 5).

Unfortunately, the safety violation still occurs when the lamps fail short-circuited (row 8). The safety assessment process reveals that if any red lamp fails short-circuited, a current of 6.0A is drawn from PS_2 (see Tab. I), and the relay coil RL_2 is again deceived. This design flaw cannot be fixed by simply adjusting the electrical parameters of the system, but requires the upgrade of the entire design as shown in Fig. 6. In the system upgrade, the additional relay coil RL_3 is *Drawn* when the current I_{PS_2} exceeds the threshold of 4.5A, that makes its contact RL_3 open, thus preventing the green lamp from turning on if a red lamp is short-circuited.

Need of quantitative modeling for verification: we make a small digression to report the main limitations we encountered

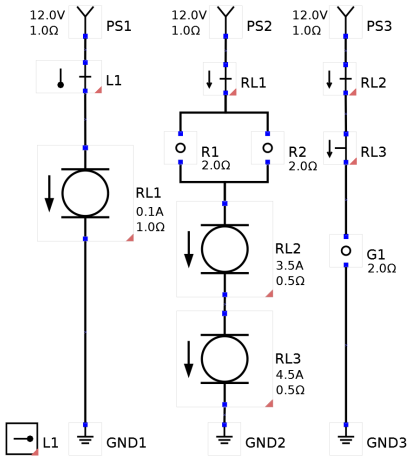

Fig. 6: Upgraded design of the RIS R2G1 from Fig. 2

while applying the traditional Boolean modeling approach (i.e. the one based on the concept of conductive paths) that led us to this work. Referring to the upgraded R2G1 design of Fig. 6, Fig. 7 shows the value of the current I_{G_1} flowing through the green lamp G_1 as a function of the current I_{PS_2} sensed by the relay coils RL_2 and RL_3. Our physical modeling approach (Fig. 7-(2)) is able to properly discriminate the faulty scenarios (i.e. $I_{PS_2} < 3.5A$ and $I_{PS_2} > 4.5A$, where $3.5A$ is the RL_2 threshold and $4.5A$ is the RL_3 threshold), keeping the green lamp properly turned-off (i.e. $I_{G_1} = 0.0A$). Differently, the expressiveness of the Boolean approach ((Fig. 7-(1))) cannot discern between different values that are greater than zero. This means that, for every current $I_{PS_2} > 0.0A$, the relay coils RL_2 and RL_3 would be considered always Drawn, resulting in a spurious behavior with the green lamp always turned-off.

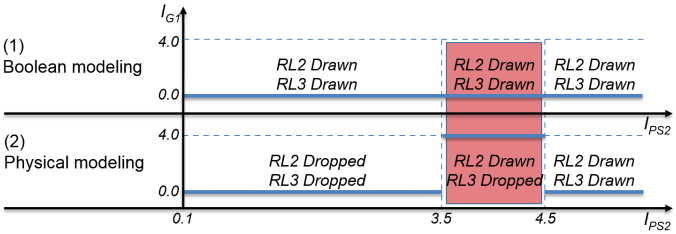

Fig. 7: Spurious behavior on the green lamp G_1 introduced by the Boolean modeling. The relay coils RL_2 and RL_3 are permanently Drawn, and keep G_1 always turned off.

VI. TOOL CHAIN

The proposed methodology was implemented in a tool chain composed of various blocks. The first block is a graphical front end (Fig. 8) based on a customization of the DIA [6] modeling environment. The palette of the front end supports over 100 distinct graphical symbols, corresponding to a subset of the components that can be found in RIS according to the Italian regulation. Each symbol is associated to an internal data structure, where parameters of various kinds are associated (e.g. delay in response time, resistance, and angular velocity).

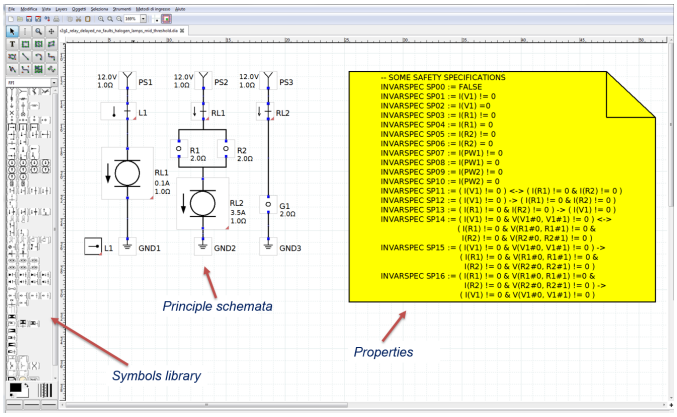

Fig. 8: Front end of our design tool.

The front end supports the connection between components, and carries out a number of sanity checks to pinpoint errors such as dangling terminals, missed components in logical connections, and conflicting logical connections between incompatible symbols. The front end also supports the definition of railway predicates representing some relevant physical conditions. Properties are expressed in form of linear temporal logic over both railway and physical predicates.

The second block is a compiler from SMDKN to hybrid automata network, symbolically expressed in the HYDI language [7]. The compiler is written in Python, and implements the conversion traversing the network based on an extensible library of behavioral component descriptions.

The third block is the HYCOMP model checker [8], that processes the resulting HYDI network and carries out the required analyses, leveraging various SMT-based engines for model checking [9], together with XSAP [10] for safety analysis and fault-trees production.

VII. EXPERIMENTAL EVALUATION

Benchmarks: we evaluated the proposed methodology analyzing a scalable, industrial-size RIS referred to as RISCS. Fig.9 shows a simplified layout of the RISCS, omitting both the electrical connections among devices and other confidential details of the relay logic. The $\text{RISCS}_{[i]}$ system represents a railway section along a bidirectional train line containing a sequence of i level crossings, with $1 \leq i \leq 10$. The section is protected on each track side by a warning and a protection semaphore. The warning/protection semaphores have three yellow/red lamps (WYL/PRL) and two green/green lamps (WGL/PGL). The lamps of the same color are electrically connected in parallel to improve the redundancy of each semaphore. Every level crossing is protected on each street side by a barrier (LCB) and by a vehicular semaphore consisting of one red lamp (LCL). The presence of the train along the line is detected by means of the train approaching pedals (TAP) and of the train detection pedals (TDP). The maintainers can completely/partially disable the section acting on several maintenance levers (GML, TAML, LCML) at the maintenance place. The train dispatcher can activate the section acting on the section enabling lever (SEL) at the train station. The relay logic is electrically connected to all the devices shown in Fig.9. The relays sense the electrical currents flowing through every connected device and actuate a specific control sequence, transferring energy between the devices. For instance, when the train pushes the left train approaching pedal (left TAP), closing its sub-circuit, the logic checks the magnitude of the current flowing through the level crossing lamps (up/down LCL) of the vehicular semaphores, and, if all the lamps work properly, the logic powers on the engines of the barriers (LCB) to start the lowering sequence.

We modeled the RISCS case studies with our tool, selecting and modeling the components and their parameters, their interconnections, and verifying properties of interest. The overall modeling task lasted for about 3 weeks, including the creation of a reusable behavioral component library.

The largest system $\text{RISCS}_{[10]}$ contains 141 power supplies, 22 resistors, 113 relays, 15 levers, 12 pedals, 678 contacts, 40 lamps, 23 maintenance lights, and 54 circuit breakers (printed on twenty A4-sheets of paper). These components are distributed over 125 sub-circuits. The conversion of the corresponding SMDKN into hybrid automaton returns an SMT encoding that uses 437 Boolean variables to encode the discrete part, and 6281 real-valued variables to encodes the physical part. Clearly, the size of the state-space makes traditional manual inspection extremely time-consuming, expensive, and unfeasible in practice.

We presents the results of the analysis on the nominal and faulty variants of the RISCS system, where up to 80 electrical faults (i.e. blown or short-circuited lamp) are injected on the 40 semaphore lamps in the case of the $\text{RISCS}_{[10]}$ benchmark.

Verification: we model checked the RISCS system against 190 invariant properties, running the two verification algorithms IC3 [11] and BMC [12] that represent complementary techniques to either verify or falsify properties. We run the experiments on a 3.5 GHz cpu with 16GB RAM, with time out (TO) set to 3600 seconds. About half of the properties represent scenarios that are supposedly feasible, and are used to validate the system design. The first validation round reported that some scenarios were found to be (unexpectedly) unfeasible. Upon fixing some buggy components in the behavior library, all the scenarios were proved to be feasible, within the timeout of 3600s, in both the nominal and faulty case. The resulting execution traces were analyzed and validated by the domain experts. Examples of scenario include that every lamp of every semaphore can be turned on and then off, or that every barrier can be completely lowered and then raised.

The remaining properties express the absence of safety violations. Most of them are verified in the nominal case within the timeout, except for three properties on the synchronization among the warning and protection semaphores.

Some relevant properties expressing the proper synchronization between the semaphore lights and the barriers positions hold also under the non-nominal case (i.e. when components are subject to faults). For instance, the model guarantees that the green lamps of the protection semaphores are off when the

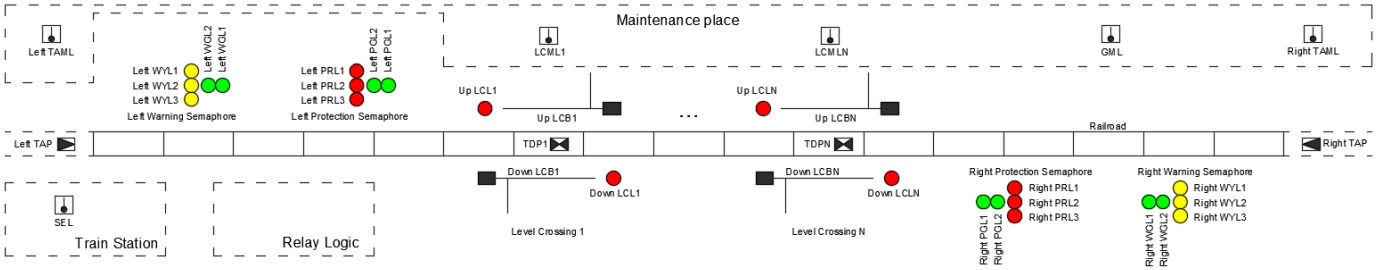

Fig. 9: Physical layout of the RISCS$_{[i]}$ case study. **Legend:** Warning Yellow Lamp (WYL), Warning Green Lamp (WGL), Protection Red Lamp (PRL), Protection Green Lamp (PGL), Level Crossing Lamp (LCL), Level Crossing Barrier (LCB), Train Approaching Pedal (TAP), Train Detection Pedal (TDP), Level Crossing Maintenance Lever (LCML), Train Approaching Maintenance Lever (TAML), General Maintenance Lever (GML), Section Enabling Lever (SEL).

level crossing barriers are not completely closed. Moreover, we are guaranteed that the colors of every semaphore are turned on in a mutually-exclusive way. Noteworthy, we successfully verified an electrical safety requirement (a low-level electrical property) prescribed by the national regulation: the level crossing lamps are short-circuited when the barriers are open and resting to prevent inadvertent activation.

59 safety properties were violated in the faulty case. Some of them check for each semaphore if there is always at least one lamp turned on. Of course, in case of multiple lamp faults, this condition cannot be avoided because all the lamp might fail. With safety analysis, we compute the fault tree responsible for the violations. For a warning semaphore, the fault tree shows that the violation might be reached in 7 distinct circumstances: either all yellow lamps are blown, or all green lamps are blown, or at least one yellow lamp is short-circuited, or at least one green lamp is short-circuited. The first two circumstances represent fault configurations of size 3 and 2, respectively the number of yellow and green lamps, that would be hard to spot by manual inspection.

VIII. Related work

Formal methods have been heavily applied in the railway domain. Important works on the verification of interlocking systems include (but are not limited to) [13], [14], [15], [16], [17], [18]. These works are not related, since they do not consider the specific case of relay circuits.

To the best of our knowledge, no works address the verification problem of a RIS based on its hybrid physical behavior. Closely related works are [19], [20], [21], [22]. While we model the evolution of continuous signals over time, the above works model Boolean signals evolving over discrete time. Furthermore, these works assume that the interaction with the environment is limited to one input per cycle to ensure that the internal micro-sequence of relay commutations started from an input command is fully extinguished (run to completion) before the arrival of the next input. In [22], two interesting observations are made. First, the discrete model of time does not support reasoning about relative time distances (e.g., between events, and on parasitic delays); second, the restriction on the number of inputs per execution cycle only works under the assumption that the control logic reacts "quickly enough" to every change in its environment. Our approach overcomes both limitations adopting a continuous model of time and not imposing restrictions on the environment. Thus, we deal with an arbitrary number of concurrent inputs and analyze the effect of inputs received in the middle of an internal micro-sequence.

We now analyze these works in more detail. The works [19], [20] present a practical approach to the RIS safety certification. A *Boolean* model is extracted from the RIS and analyzed via SAT-based abstraction-refinement. Our SMT-based approach enables more fine grained analyses, modeling the precise physics of the system and preventing spurious behaviors introduced by the Boolean abstraction. The work [21] builds a Boolean model based on the abstraction concept of *conductive path*: a relay coil is drawn iff all the conduction conditions along a conductive path from a power supply to the coil are satisfied. This approach is subject to several limitations: it is only valid under some assumptions on the system physics (e.g., all the power supplies are always up and running); it requires the enumeration of a potentially exponential number of conductive paths; it does not permit a quantitative reasoning (e.g., how much current flows through a conductive path). There is only one work [22] that considers risk analysis and the effects of single-mode faults on the system safety. These faults are Boolean and limited to the discrete state of relays (e.g., stuck at dropped/drawn). In our work we allow the designer to specify a larger class of faults, both on the discrete and physical state of components, with no limitation on the contemporaneity of fault occurrences.

IX. Conclusion

In this paper we proposed an approach to understand legacy relay circuits in the railway domain. We rely on an accurate representation at the physical level in form of Switched Kirchhoff Networks, that is then reduced to a symbolically represented network of hybrid automata, and then analyzed by means of SMT-based model checking. The experimental evaluation demonstrates the precision and scalability of the analyses. The proposed methodology is at the core of an ongoing research project aiming at the in-the-large analysis of legacy railway interlocking and the open specification of computer-based solutions. Directions for future research include the definition of a library of property patterns, the definition of specific verification engines, and the integrated animation of counterexamples.

REFERENCES

[1] A. Cimatti, S. Mover, and M. Sessa, "SMT-based analysis of switching multi-domain linear Kirchhoff networks," in *2017 Formal Methods in Computer Aided Design, FMCAD 2017*, 2017, pp. 188–195.

[2] Comitato Elettrico Italiano, "UNIFER-CEI S 461: sigle e segni grafici per gli schemi dei circuiti elettrici degli impianti di segnalamento ferroviario," Comitato Elettrico Italiano, Standard, 1976.

[3] A. Cimatti, S. Mover, and M. Sessa, "From Electrical Switched Networks to Hybrid Automata," in *FM 2016: Formal Methods, Proceedings*, 2016, pp. 164–181.

[4] T. A. Henzinger, "The theory of hybrid automata," in *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science, 1996*, 1996.

[5] P. Fritzson, *Principles of object-oriented modeling and simulation with Modelica 3.3: a cyber-physical approach*. John Wiley & Sons, 2014.

[6] GNOME, "Dia," https://gitlab.gnome.org/GNOME/dia, 2017.

[7] A. Cimatti, S. Mover, and S. Tonetta, "Hydi: A language for symbolic hybrid systems with discrete interaction," in *37th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2011*. IEEE Computer Society, 2011, pp. 275–278.

[8] A. Cimatti, A. Griggio, S. Mover, and S. Tonetta, "HyCOMP: An SMT-based model checker for hybrid systems," in *TACAS*, 2015.

[9] R. Cavada, A. Cimatti, M. Dorigatti, A. Griggio, A. Mariotti, A. Micheli, S. Mover, M. Roveri, and S. Tonetta, "The nuXmv symbolic model checker," in *CAV*, ser. Lecture Notes in Computer Science, vol. 8559, 2014, pp. 334–342.

[10] B. Bittner, M. Bozzano, R. Cavada, A. Cimatti, M. Gario, A. Griggio, C. Mattarei, A. Micheli, and G. Zampedri, "The xSAP safety analysis platform," in *Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2016, Proceedings*, 2016, pp. 533–539.

[11] A. Cimatti, A. Griggio, S. Mover, and S. Tonetta, "Infinite-state invariant checking with IC3 and predicate abstraction," *Formal Methods in System Design*, vol. 49, no. 3, pp. 190–218, 2016.

[12] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, and Y. Zhu, "Bounded model checking," *Advances in Computers*, vol. 58, pp. 117–148, 2003.

[13] A. E. Haxthausen and P. H. Østergaard, "On the use of static checking in the verification of interlocking systems," in *Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications - 7th International Symposium, ISoLA 2016, Proceedings, Part II*, 2016, pp. 266–278.

[14] L. V. Hong, A. E. Haxthausen, and J. Peleska, "Formal modelling and verification of interlocking systems featuring sequential release," *Sci. Comput. Program.*, vol. 133, pp. 91–115, 2017.

[15] A. Fantechi, A. E. Haxthausen, and H. D. Macedo, "Compositional verification of interlocking systems for large stations," in *Software Engineering and Formal Methods, SEFM 2017, Proceedings*, 2017, pp. 236–252.

[16] V. Hartonas-Garmhausen, S. V. A. Campos, A. Cimatti, E. M. Clarke, and F. Giunchiglia, "Verification of a safety-critical railway interlocking system with real-time constraints," *Sci. Comput. Program.*, vol. 36, no. 1, pp. 53–64, 2000.

[17] A. Cimatti, R. Corvino, A. Lazzaro, I. Narasamdya, T. Rizzo, M. Roveri, A. Sanseviero, and A. Tchaltsev, "Formal verification and validation of ERTMS industrial railway train spacing system," in *Computer Aided Verification, CAV 2012, Proceedings*, 2012, pp. 378–393.

[18] A. Cimatti, F. Giunchiglia, G. Mongardi, D. Romano, F. Torielli, and P. Traverso, "Formal verification of a railway interlocking system using model checking," *Formal Asp. Comput.*, vol. 10, no. 4, pp. 361–380, 1998.

[19] A. Bonacchi, A. Fantechi, S. Bacherini, and M. Tempestini, "Validation process for railway interlocking systems," *Sci. Comput. Program.*, vol. 128, pp. 2–21, 2016.

[20] A. Bonacchi, A. Fantechi, S. Bacherini, M. Tempestini, and L. Cipriani, "Validation of railway interlocking systems by formal verification, A case study," in *Software Engineering and Formal Methods - SEFM 2013*, 2013, pp. 237–252.

[21] A. E. Haxthausen, A. A. Kjær, and M. L. Bliguet, "Formal development of a tool for automated modelling and verification of relay interlocking systems," in *FM 2011: Formal Methods, Proceedings*, 2011, pp. 118–132.

[22] L. Eriksson, "Using formal methods in a retrospective safety case," in *Computer Safety, Reliability, and Security, SAFECOMP 2004, Proceedings*, 2004, pp. 31–44.