A Framework for the Formal Specification of Relay-Based Systems Based on a b-Method Graph Specification

Dalay Israel de Almeida Pereira[1*], Matthieu Perin[2], Philippe Bon[1], Simon Collart-Dutilleul[1]
[1] Univ Lille Nord de France, IFSTTAR, COSYS/ESTAS, 59650 Villeneuve d'Ascq, France.
[2] Institut de Recherche Technologique Railenium, F-59300 Famars, France.

* Corresponding author. Tel.: +33 (0) 3 20 43 84 07; email: dalay-israel.de-almeida-pereira@ifsttar.fr
Manuscript submitted September 14, 2018; accepted December 10, 2018.
doi: 10.17706/ijcee.2019.11.1.11-19

Abstract: A railway interlocking system is one example of a critical system, and, therefore, it must have a high level of reliability in order to avoid problems that may result on the loss of people's lives. However, many railway systems are still specified using historical relay-based diagrams, whose analysis are made by human inspection, which is error prone. Relay-based diagrams are specified by nodes and cables in a graphical manner, which resemble undirected graphs. This paper presents a framework for the specification of relay diagrams in a formal language, B-method, based on the specification of a graph and its properties. The use of a formal language allows one to prove the correctness of these railway interlocking systems regarding structural properties. This framework has been evaluated by the specification of a case study.

Key words: Railway interlocking systems, relay diagrams, graph, B-method, framework.

1. Introduction

The use of increasingly complex applications is demanding a greater investment of resources in system specification. Furthermore, high field reliability is a highly desirable attribute of any product, system or plant [1]. Railway Interlocking systems (RIS) is an example of critical systems, where reliability may be the differentiating factor determining its success or the occurrence of significant negative consequences (like the loss of people's lives, for instance). These systems must have strict requirements for security and safety, in order to protect the user [2]. In this case, the use of advanced modelling techniques in order to model these RIS may be a crucial factor in order to improve safety and reliability.

Despite the demand for new methodologies of specification, many railway systems are still specified by historical relay-based diagrams, which describes how the physical elements (nodes) of these systems are connected by cables. These diagrams are graphs where the nodes and cables are, respectively, the vertices and edges of the graph. These systems are usually described without any explicit formalisation. Furthermore, in order to analyse their correctness, experts must inspect the diagrams and draw conclusions, which may not be satisfactory for critical systems, since it is error prone [3]. Furthermore, a language that has gaining space on the railway system specification is B-method [4]. With the aim of "analyse, study and specify, not only software, but also whole systems" [5], this specification language has been successfully applied in railway projects ([6], [7]).

Based on these facts, this paper proposes a framework for the formal specification of relay diagrams using B-method. By taking as basis the specification of a general undirected graph, it is possible to describe

a railway control system and verify its structural correctness based on the logic provided by B and the graph properties. Besides, as future contribution, we aim to use the specification refinement supported by the B-method in order to implement these historical systems as computer-controlled systems.

Some works described graphs and railway interlocking systems in B. The work presented in [8] is one example of the use of B-method in order to specify graphs. However, its graph specification is associated to a Petri-net, which is different of our general undirected graph definition. Similar ideas may be found in [9]-[12]. Furthermore, The work presented in [9] applies this specification to an example of the railway domain, focused on the behaviour of the system described in Petri-nets. In our work, we present a method for the formal specification of the physical structure of relay-based systems (usually specified by relay-based diagrams), which is commonly ignored in the railway systems formal specification methods existing in the literature. The objective of this method is to verify the correctness of structural properties of these systems.

Furthermore, other works (like [13] and [14]) used formal methods in order to specify and verify railway computer controlled interlocking systems. However, relay-based systems are still used in the railway domain. In order to transform these systems into computer-controlled systems, the industry needs easy and reliable ways to make this transformation maintaining the functionality of the system. This work is a first step towards the specification of relay-based systems in B as a way to refine the specifications in order to implement them as computer controlled systems.

The next section of this paper (Section 2) presents the background of this work, which is comprised by some notions of Relay-based Specification, B-methods and Graphs. Section 3 details our graph specification using B-method, which is the basis for the specification of relay-based diagrams in B, as presented in Section 4. Section 5 discusses about the use of this framework in a case study and Section 6 concludes the paper and presents the next steps of this work.

2. Formalisms Used

2.1. Relay-Based Specification

In the railway domain, in an environment where the trains are not computer-controlled, the transmission, reception and use of information inside the system are made by electromechanical relays [15]. These objects are composed by an electromagnet and one or more mobile mechanical parts (contacts). A contact may connect different wires by changing its position when affected by the electromagnet inside a relay. The different combinations of different nodes (power sources, contacts and relays, for instance) allow the creation of several different behaviours.

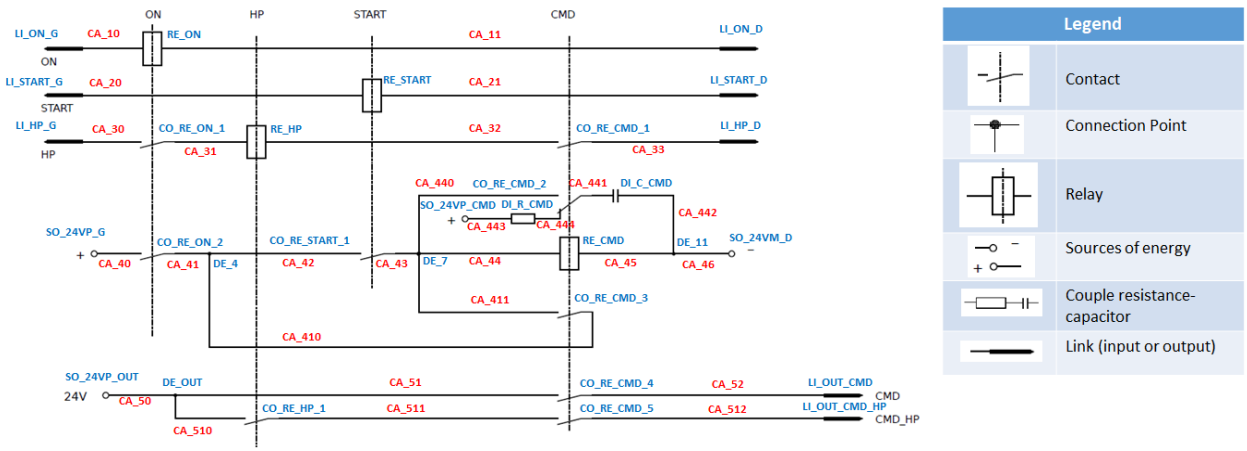


Fig. 1. Example of a relay-based diagram.

The Fig. 1 is one example of relay-based diagram which presents the specification of the initial part of a command system that can be used in industry. This small part of the system aims to guarantee a safe shutdown of the system by avoiding the transition from a high powered state to an off state. The safe shutdown allow the system to stay in a normal powered state for a small period of time, which may, for instance, allow it to cool down. This diagram is used as a running example throughout this paper.

In order to describe relay-based systems, relay-based diagrams are a standard tool. These diagrams are drawings presenting all the nodes of a system and their interconnections. In fact, the relay-based diagrams are undirected graphs. Two nodes (vertices) are connected when there is a cable (edge) that makes contact with each of these nodes. The cables have an explicit and unequivocal syntax: it physically connects the nodes of the system in order to allow the flow of electric current. Furthermore, there is another relation between the nodes, more specifically, between contacts and relays denoted by a vertical mixed fine line. This last notation represents the functional and mechanical link between a relay and its associated contacts, that is, the electromagnetic influence of the relay over the contacts.

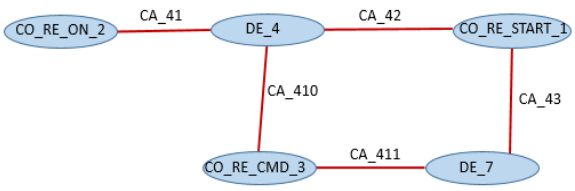
In the railway industry, a project of a railway control system specified by relay diagrams contains several sheets of specification. Each sheet describes a small part of the system and contains a diagram similar to the one presented in Fig. 1. Two diagrams in two different sheets are connected by their inputs and/or outputs and the union of all sheets describes the whole behaviour of a control system.

2.2. B-Method

According to [4], B is a method for specifying, describing and coding software systems. By making use of a strong mathematical background, it allows the specification and verification of systems in a formal manner in order to guarantee a high level of reliability. The first successful use of this method in an industrial case was the specification of the Meteor line 14 drive-less metro [7], in Paris. Since then, other systems has been successfully specified and implemented using B-method, like, for instance COPPILOT [5]. Besides B has been also used for proving the correctness of systems, like, for instance, SACEM [6]

The basic building block of a B-method specification is the abstract machine [16]. One system may be specified by one or several machines. The specification inside a machine is divided in many parts, each one under an appropriate heading (or clause) describing a different aspect of the specification. The first heading, “MACHINE”, starts the specification of an abstract machine, whose name must be described under this heading.

The local state of a machine is kept by the variables which are defined under the “VARIABLES” clause and whose details are defined under the “INVARIANT” heading. These details comprises variables typing and properties that must be satisfied in the specification. The initial state of the machine must be described in the “INITIALISATION” clause. It is also possible to describe constant information, like, for instance, sets of information that can be used inside the machine, which are described under the “SETS” heading.




```
MACHINE Graph
SETS VERTICES = {CO_RE_ON_2, DE_4, CO_RE_START_1,
                 CO_RE_CMD_3, DE_7};
     EDGES = {CA_11, CA_42, CA_43, CA_410, CA_411}
VARIABLES vertices, edges
INVARIANT vertices <: VERTICES & edges <: EDGES
INITIALISATION vertices := {CO_RE_ON_2, DE_4, CO_RE_START_1,
                            CO_RE_CMD_3, DE_7}
           || edges := {CA_11, CA_42, CA_43, CA_410, CA_411}
END
```

Fig. 2. Simple machine example described in B.

An example of a simple B-machine is presented in Fig. 2. This machine specifies the sets of vertices and

edges of a graph, which is a part of the relay diagram presented in Fig. 1. The universal sets of vertices and edges are described in the "SETS" clause, named as "VERTICES" and "EDGES", respectively. The machine also defines the variables "vertices" and "edges", which are subsets of the universal sets, and initialised with their elements. The notation "<:" is the ASCII version for "subset" (⊆). A list of all the ASCII notations, which are used throughout this paper, can be found in [16].

In order to extend a machine, one may use the "EXTENDS" heading, followed by the name of the machines one wants to extend. The "SEES" heading allows one machine to see the values of constants and variables of other machines. Although there are more clauses that may be used inside a machine, for sake of brevity they will not be discussed in this paper. More information about these and other clauses are presented in [16].

2.3. Graphs

On its most basic definition, a graph is an ordered triple $(V(G), E(G), \varphi_G)$, consisting of a non-empty set of vertices $V(G)$, a set of edges $E(G)$ and an incidence function φ_G from the set of edges to unordered pairs of vertices from $V(G)$ [17]. The Fig. 2 depicts an example of a graph. In this example, considering the nodes and cables of the diagram as the vertices and edges of a graph G, respectively, the sets $V(G)$, $E(G)$ and φ_G are:

- $V(G) = \{CO\_RE\_ON\_2, DE\_4, CO\_RE\_START\_1, CO\_RE\_CMD\_3, DE\_7\}$
- $E(G) = \{CA\_11, CA\_42, CA\_43, CA\_410, CA\_411\}$
- φ_G = {(CA_41, (CO_RE_ON_2, DE_4)), (CA_42, (DE_4, CO_RE_START_1)), (CA_43, (CO_RE_START_1, DE_7)), (CA_410, (DE_7, CO_RE_CMD_3)), (CA_411, (CO_RE_CMD_3, DE_4))}

Several properties can be used to describe and differentiate graphs. The number of connections of a vertice is the vertice degree. A path of a graph is a sequence of vertices in a way that each element of the sequence is connected to its previous element by an edge inside the graph. Besides, all the vertices of the sequence are different (with no repetition). Based on this, two vertices are connected inside a graph if there is a path connecting them. So, considering the graph on Fig. 2, *DE_4* has a vertice degree 4 and the sequence *<DE_4, CO_RE_START_1, DE_7, CO_RE_CMD_3>* is a path that connects the vertices *DE_4* and *CO_RE_CMD_3*.

3. Graph Specification in B

B comports the description of sets and functions in a simple manner, which allows the description of the basic graph definition that has been presented on Section 2.3. In order to specify the elements of a graph (vertices and edges) in B, one must describe the universal sets of elements. These sets can be specified under the "SETS" clause, as it was presented in Fig. 2. Based on the universal sets, it is possible to define the triple $(V(G), E(G), \varphi_G)$ by creating three variables: the sets "vertices" and "edges", representing the sets of vertices and edges of a graph, and an incidence function "incidence" from the set of edges to unordered pairs of vertices. Fig. 3 presents the graph specification of our running example, which exemplifies how one may represent the graph triple.

The sets "vertices" and "edges" are defined as subsets of the universal sets "VERTICES" and "EDGES", respectively. Furthermore, based on the definition of a graph, the "incidence" function is specified as a total function from "edges" to pairs of vertices, which is represented in B by the cartesian product "vertices*vertices".

Then, B can be used in order to impose restrictions or analyse a graph specification. For instance, one may describe the degree of a vertice *vv*, based on the quantity of times this vertice is related to itself or to another vertice. This property can be specified in B by the sum: "card({vv}<|ran(incidence)) + card(ran(incidence)|>{vv})". Based on the definition of properties like this, one can define, for instance, an invariant that limits the vertice degree to 1: "!(vv).((vv:vertices) => card(vv<|ran(incidence)) + card(ran(incidence)|>vv) <= 1)".

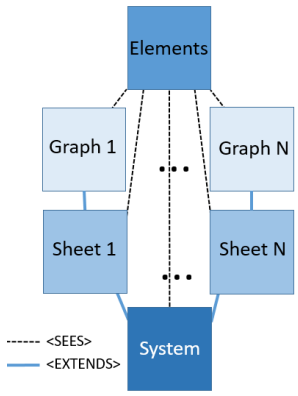
Another example of property that can be described in B is the concept of path. On its most basic definition, a path is a sequence of different vertices, that is, “path : iseq(vertices)”. However, it is necessary to assure that each element is connected to its successor in the sequence: “!(xx, yy).((xx:dom(path) & yy:ran(path) & (xx, yy):path & xx<size(path)) => ((path(xx+1),yy) : ran(incidence)”.

```
MACHINE undirectedGraph
SEES Elements
VARIABLES vertices, edges, incidence
INVARIANT vertices <: VERTICES &  edges <: EDGES & incidence : edges --> (vertices*vertices)
INITIALISATION vertices := {CO_RE_ON_1, CO_RE_ON_2, CO_RE_HP_1, CO_RE_START_1, CO_RE_CMD_1,
   CO_RE_CMD_2, CO_RE_CMD_3, CO_RE_CMD_4, CO_RE_CMD_5, LI_ON_D, DE_7, DE_11, DE_OUT, DI_R_CMD,
   DI_C_CMD, LI_ON_G, SO_24VP_OUT, DE_4, LI_START_G, LI_START_D, LI_HP_G, LI_HP_D, LI_OUT_CMD,
   LI_OUT_CMD_HP, RE_ON, RE_HP, RE_START, RE_CMD, SO_24VP_CMD, SO_24VP_G, SO_24VM_D, LI_ON_D}
|| edges := {CA_10, CA_11, CA_20, CA_21, CA_30, CA_31, CA_32, CA_33, CA_40, CA_41, CA_42,
   CA_43, CA_44, CA_45, CA_46, CA_410, CA_411, CA_440, CA_441, CA_442, CA_443, CA_444, CA_50,
   CA_51, CA_52, CA_510, CA_511, CA_512}
|| incidence := {CA_10 |-> (RE_ON, LI_ON_G), CA_511 |-> (CO_RE_HP_1, CO_RE_CMD_5),
   CA_20 |-> (LI_START_G, RE_START), CA_21 |-> (RE_START, LI_START_D), CA_46 |-> (DE_11, SO_24VM_D),
   CA_30|->(LI_HP_G, CO_RE_ON_1), CA_31 |-> (CO_RE_ON_1, RE_HP), CA_32 |-> (RE_HP, CO_RE_CMD_1),
   CA_33 |-> (CO_RE_CMD_1, LI_HP_D),CA_40|->(SO_24VP_G, CO_RE_ON_2), CA_41 |-> (CO_RE_ON_2, DE_4),
   CA_42 |-> (DE_4, CO_RE_START_1), CA_42 |-> (DE_4, CO_RE_START_1), CA_11 |-> (RE_ON, LI_ON_D),
   CA_45 |-> (RE_CMD, DE_11), CA_43 |-> (CO_RE_START_1, DE_7), CA_51 |-> (DE_OUT, CO_RE_CMD_4),
   CA_441 |->(CO_RE_CMD_2, DI_C_CMD), CA_442 |-> (DI_C_CMD, DE_11), CA_410 |-> (DE_4, CO_RE_CMD_3),
   CA_411 |-> (CO_RE_CMD_3, DE_7), CA_510 |-> (DE_OUT, CO_RE_HP_1), CA_50 |-> (SO_24VP_OUT, DE_OUT),
   CA_44 |-> (DE_7, RE_CMD), CA_512 |->(CO_RE_CMD_5, LI_OUT_CMD_HP), CA_440 |-> (DE_7, CO_RE_CMD_2),
   CA_52 |-> (CO_RE_CMD_4, LI_OUT_CMD)}
END
```

Fig. 3. Graph specification in B of the safe shutdown example.

4. Framework for the Specification of Relay-Based Systems in B

This section presents the framework for the specification of relay-based diagrams in B, which uses a graph specification as basis. The framework consists of four different kinds of B machines that can be used to specify a relay-based diagram and guarantee that it meets some structural properties. The first two kinds are related to the graph specification, as presented in the Section 3, however, here we separate the “SETS” clause in a machine “Elements” and the graph in a machine “Graph”. The third machine kind describes details of a sheet of specification and the last machine describes the complete system by extending all the sheets and making the links between their inputs and outputs. In a specification of a system, one may have many sheets and, therefore, many graphs. However, a system always have one unique universe of elements (that can be seen by all other machines) and one unique instance of the complete system. Fig. 4 presents the relation between all these machines.




```
MACHINE
    Elements
SETS VERTICES = {Node_1, Node_2 ... Node_N};
    EDGES = {Cable_1, Cable_2 ... Cable_N}
END
```

```
MACHINE Graph
SEES Elements
VARIABLES vertices, edges, incidence
INVARIANT vertices <: VERTICES &  edges <: EDGES &
          incidence : edges --> (vertices*vertices)
INITIALISATION vertices := {Node_1, Node_2 ... Node_N}
   || edges := {Cable_1, Cable_2 ... Cable_N}
   || incidence := {Cable_1 |-> (Node_1, Node_2),
                    Cable_2 |-> (Node_2, Node_3) ...
                    Cable_N |-> (Node_N-1, Node_N)}
END
```

```
MACHINE Sheet
SEES Elements;
EXTENDS Graph
VARIABLES cables, type_1, type_2, ... ,type_N
INVARIANT cables = edges & type_1 <: vertices &
    type_2 <: vertices & ... & type_N <: vertices
INITIALISATION cables := edges ||
    type_1 := {Node_1, Node_2, ... ,Node_M} ||
    type_2 := {Node_3, ... , Node_O} || ... ||
    type_N := {Node_N, ... , Node_P}
END
```

```
MACHINE
    System
EXTENDS Sheet1.Sheet_1, Sheet2.Sheet_2,
        ... ,SheetN.Sheet_N
VARIABLES folioLiaisons
INVARIANT func : VERTICES <-> VERTICES
INITIALISATION func := {Link1 |-> Link2, ...}
END
```

Fig. 4. Framework for the specification of relay-diagrams in B.

Generally, a relay diagram consists of many different nodes (*Node_1, Node_2, ..., Node_N*) connected by cables (*Cable_1, Cable_2, ..., Cable_N*). Each node has a type (relay, contact or source of energy, for instance), which may be equal or different from the type of another node. Based on this general description, we can specify the machines that constitutes our framework, which are depicted on Fig. 4.

The content of the machines "Elements" and "Graph" is the graph specification that have been discussed in Section 3. The former machine contains the specification of the universal sets of vertices and edges (nodes and cables, respectively) of the specification. The latter presents the graph based on the triple presented in [17], but the variables are initialised in order to specify the diagram of a sheet of the system. By dividing this specification into two different machines we aim to create an isolate machine for the universal sets that can be seen by all the other machines.

In a relay-based specification, a sheet is more than a graph, since it contains more information about the elements (like the type of each node and the relation between contacts and relays, for instance). So, the machine "Sheet" describes details of a sheet of specification. Initially it distinguishes all the nodes into subgroups according to their types. Furthermore, this machine must contains several other details, like:

- "Relay_Activ : contacts --> relays", which specifies the relation between the contacts and relays in a variable "relay_Activ";
- "Link_Type: links --> {input, output}", which defines each link as an input or output of the sheet in a variable "link_type";
- "SourceType : sources --> {plus_24, minus_24, in_400Hz, out_400Hz}", which specifies a relation between the sources of energy and their types.

Besides, one may specify invariants in order to verify the structural correctness of the system. Some examples of structural properties that can be verified are:

- The limit of cables connected to each node based on the vertice degree (for instance, sources of energy must be connected to only one cable),
- The existence of a path between an output and a source of energy with the aim of guaranteeing that the output can send electrical information,
- The non-existence of a path between sources of energy of different types.

The last machine, "System", concludes the framework by extending all the sheets and connecting the links between them through a relation. Invariants can also be described in this machine in order to guarantee that the relation image and domain only connects links of any sheet.

5. Case Study

The "safe shutdown" example, which was used throughout this paper, has been specified using our framework and some details and results regarding the B specification of the example is presented in Table 1. In order to analyse the machines we use Atelier B, which is an industrial tool for the specification, refinement and implementation of systems using B-method.

Table 1. Case Study Details and Results

Number of nodes	31
Number of cables	28
Obligation proofs	27
Proved obligation proofs	27
Unproved obligation proofs	0
Proved structural properties	9

As it is presented in the Table 1, Atelier-B generated 27 obligation proofs (OPs) in order to prove the consistence of the "Sheet" machine, which has been completely proved. From these 27 OPs, 9 of them

regards to the proof of structural properties, which are organised as follows:

- 2 proofs in order to assure that each node has a (single) type,
- 6 proofs guarantees the right number of connections of nodes to cables and
- 1 proof that guarantees that each output is connected to at least one source of energy.

6. Conclusion

This work presents a framework for the specification of relay-based diagrams using B-method. This specification is composed by four different machines with different objectives: the specification of the universal sets of cables and nodes, the specification of the graph that represents a relay diagram, the specification of a sheet of a relay diagram (containing details about the nodes) and the specification of the complete system composed by all the sheets. By specifying the structure of the relay diagram by a graph, it is possible to define structural properties that the system must satisfy.

This framework has been used in order to specify an example of relay-based diagram that can be used in industry. As a result, regarding the specification of a relay-based diagram sheet all the 27 obligation proofs have been proved, which includes 9 proofs regarding structural properties of the relay-based diagram.

In our near future agenda, we aim to specify more relay-based structural properties in B in order to improve the quality of the specification. Furthermore, the next step of our work is to make a behavioural analysis of the relay-based specification based on the inputs and outputs of the system. We aim to use this analysis in order to improve the capabilities of the framework and to refine the B specification towards its implementation. Besides, we aim to specify and implement an automatic transformation from the relay-based diagrams towards B-method, based on our framework and using the model defined in [18] as the transformation input.

References

[1] Barnard, R. W. A. (2008, June). What is wrong with reliability engineering? *Proceedings of the INCOSE International Symposium: Vol. 18, No. 1.* (pp. 357-365).

[2] Hinchey, M., & Coyle, L. (2010, March). Evolving critical systems: A research agenda for computer-based systems. *Proceedings of the 2010 17th IEEE International Conference and Workshops on Engineering of Computer Based Systems (ECBS)* (pp. 430-435).

[3] Haxthausen, A. E. (2010, March). Towards a framework for modelling and verification of relay interlocking systems. *Proceedings of the Monterey Workshop* (pp. 176-192). Springer, Berlin, Heidelberg.

[4] Abrial, J. R., & Abrial, J. R. (2005). *The b-Book: Assigning Programs to Meanings*. Cambridge University Press.

[5] Lecomte, T., Servat, T., & Pouzancre, G. (2007, August). Formal methods in safety-critical railway systems. *Proceedings of the 10th Brasilian Symposium on Formal Methods* (pp. 29-31).

[6] Guiho, G., & Hennebert, C. (1990, February). SACEM software validation. *Proceedings of the 12th International Conference on Software Engineering* (pp. 186-191). IEEE Computer Society Press.

[7] Behm, P., Benoit, P., Faivre, A., & Meynadier, J. M. (1999, September). METEOR: A successful application of B in a large project. *Proceedings of the International Symposium on Formal Methods* (pp. 369-387). Springer, Berlin, Heidelberg.

[8] Attiogbe, C. (2009, March). Semantic embedding of Petri Nets into event-B. *Proceedings of the Integration of Model-Based Formal Methods Tools (IM_FMT@ IFM'2009)*.

[9] Bon, P., & Collart-Dutilleul, S. (2013). From a solution model to a B model for verification of safety properties. *Journal of Universal Computer Science*, *19(1)*, 2-24.

[10] Sun, P., Bon, P., & Collart-Dutilleul, S. (2015). A joint development of coloured petri nets and the B method in critical systems. *Journal of Universal Computer Science, 21(12)*, 1654-1683.

[11] Boudi, Z., Ben-Ayed, R., Collart-Dutilleul, S., Nolasco, T., & Haloua, M. (2017). A CPN/B method transformation framework for railway safety rules formal validation. *European Transport Research Review, 9(2)*, 13.

[12] Korečko, Š., & Sobota, B. (2014). Petri nets to B-language transformation in software development. *Acta Polytechnica Hungarica*, *11(6)*, 187-206.

[13] Haxthausen, A. E., Peleska, J., & Kinder, S. (2011). A formal approach for the construction and verification of railway control systems. *Formal Aspects of Computing*, *23(2)*, 191-219.

[14] Bernardeschi, C., Fantechi, A., Gnesi, S., Larosa, S., Mongardi, G., & Romano, D. (1998). A formal verification environment for railway signaling system design. *Formal Methods in System Design*, *12(2)*, 139-161.

[15] Rétiveau, R. (1987). La signalisation ferroviaire. P*resse de l'école Nationale des Ponts et Chaussées*.

[16] Schneider, S. (2001). *The B-Method: An Introduction*. Palgrave.

[17] Bondy, J. A., & Murty, U. S. R. (1976). *Graph Theory with Applications* (Vol. 290). London: Macmillan.

[18] Almeida-Pereira, D. I., Malki, O., Bon, P., Perin, M., & Collart-Dutilleul, S. (2018, October). An MDA approach for the specification of relay-based diagrams. *Proceedings of the International Conference on Model and Data Engineering* (pp. 17-29). Springer, Cham.

Dalay Israel de Almeida Pereira is a PhD student in computer science at Lille University, IFSTTAR/COSYS/ESTAS, Lille, France. In 2015 he graduated as a bachelor of software engineering in the Federal University of Rio Grande do Norte (UFRN) and, in 2017, he received the master degree in systems and computing from the same university. His research interests include formal methodologies of specification and formal specification of railway critical systems.

Matthieu Perin defended a PhD in computer science at the ENS Cachan in LURPA Lab. He was a contractual reasercher on the LCHIP projet in charge of the meta-modelling of the realy-based diagrams. He is currently a researcher at IRT Railenium, a public-private research foundation in railway innovation. He is part of the autonomous train project, responsible for the onboard model of the railway. His research area cover modelling & formal methods application to railway infrastructure, control logic and ERTMS System.

Philippe Bon received the Ph.D. degree from Lille University in 2000. He is currently a senior researcher in the ESTAS Laboratory, COSYS Department, the French Institute of Science and Technology for Transport, Development and Net-works (IFSTTAR). His research focuses on the implementation requirements traceability throughout the design cycle of railway land systems. He was involved on several research projects related to the use of formal methods for traceability and validation.

Simon Collart Dutilleul received the Ph.D. degree from University of Savoie in 1997 and the habilitation degree from Lille University in 2008. In 1999, he was a lecturer at the Lagis Laboratory, Ecole Centrale de Lille. He was responsible for the teaching section in computer engineering at the Institute of Technology of Computer and Industrial Engineering from 2006 to 2012. He is currently a research director at IFSTTAR. He has supervised nine Ph.D. students. Since 2012, he has been involved in the IFSTTAR/COSYS/ESTAS Laboratory dedicated to railway systems. He also leads the European Railway Management System Group in this laboratory.