



HAL
open science

Analysis and formal specification of relay-based railway interlocking systems

Dalay Israel de Almeida Pereira

► To cite this version:

Dalay Israel de Almeida Pereira. Analysis and formal specification of relay-based railway interlocking systems. Automatic Control Engineering. Centrale Lille Institut, 2020. English. NNT : 2020CLIL0009 . tel-03215450

HAL Id: tel-03215450

<https://theses.hal.science/tel-03215450v1>

Submitted on 3 May 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N° d'ordre : 401



CENTRALE LILLE

THÈSE

présentée en vue d'obtenir le grade de

DOCTEUR

en

Spécialité : Informatique, Automatique

par

Dalay Israel de Almeida Pereira

DOCTORAT DELIVRÉ PAR CENTRALE LILLE

Titre de la thèse :

Analyse et Spécification Formelle des Systèmes d'Enclenchement Ferroviaire Basés sur les Relais

Analysis and Formal Specification of Relay-based Railway Interlocking Systems

Soutenue le 15 Octobre 2020 devant le jury d'examen :

Président:	Jean-Paul Bodeveix, Professeur, <i>Université Paul Sabatier</i>
Rapporteur:	Catherine Dubois, Professeure, <i>École Nationale Supérieure d'Informatique pour l'Industrie et l'Entreprise</i>
Rapporteur:	Helen Treharne, Professeure, <i>University of Surrey</i>
Examineur:	Marcel Vinicius Medeiros Oliveira, Professeur, <i>Universidade Federal do Rio Grande do Norte</i>
Directeur de thèse:	Simon Collart-Dutilleul, Directeur de recherche, <i>Université Gustave Eiffel</i>
Encadrant:	Philippe Bon, Chargé de recherche, <i>Université Gustave Eiffel</i>
Encadrant:	Matthieu Perin, Ingénieur de recherche, <i>IRT Railenium</i>

Thèse préparée dans le Laboratoire d'Évaluation des Systèmes de Transports
Automatisés et de leur Sécurité
Université Gustave Eiffel, COSYS/ESTAS, Villeneuve d'Ascq
École Doctorale SPI 072 (EC Lille)

Acknowledgements

I would like to thank all those who supported me in my doctoral studies. This work could not be achieved without your assistance and encouragement.

First and foremost, I thank god, the universe and/or the fate, for the opportunity of being here and accomplishing this work. The world is something amazingly interesting and it is always an honour to have the chance of studying it.

An undoubtedly thanks goes to my parents. I would never be here if it wasn't for the strength, perseverance, kindness and patience of my mother, Denise, and father, Beto, who always gave their best for us to have everything we needed and which always taught me that I must invariably give the best of me. You'll always be my best references and I cannot just thank you enough. Besides, you have raised my sisters, Poliana and Adriana, who are some of the people that knows me better than anyone. You gave me some of the most important lessons about love and altruism that I could ever receive. You all are the proof that unconditional love exists and I'll always love you.

Clarissa, my niece, you are the fruit of a family full of love, which resulted in a child extremely kind, lovely and perfect. One of the things I am most proud in my life is taking care of you in your first two years. Although I had to leave for the doctoral studies, I never stopped thinking about you. You're my strength and one of my most important motivations to become a better person. You are my definition of love.

Paulo, my brother in law, thank you for introducing me to the IT. You were a key part of my path until here. Thank you for supporting me and being such a reference in my academic life.

I also dedicate this work specially to Cícera, Deusarina, Neto, Sylvia and Deusinha. Cícera, my grandmother and second mother, you are one of the people that raised me to be the best person I could be. I love you in a way I just can't describe. Deusarina and Neto, my aunt and uncle, my journey until here started when you selflessly helped me, my sisters and my parents to have a better quality of life and studies. This small seed is the origin of several opportunities and growth we have had. I will be always thankful to you. You are some of my greatest examples of generosity, altruism and kindness. Sylvia and Deusinha, godmother and aunt, I dedicate this thesis to you because you always believed and encouraged me in my academic journey. During these three years, you were some of the people that most spoke with me, comforting me in difficult moments, making me laugh and saying how proud you were, which makes me feel proud about the path I chose to follow. I thank god for having such a lovely family which always supports and encourages me to follow my dreams.

Yasmin, you are to me a sister that we adopted through the mutual love between my family and you. It is always so nice to have you around and I have learnt so much from you. Thank you for spending this last year with me in the most important moments of my work. You made my days to be lighter when I was stressed out. You are an incredible person, a great sister and a perfect friend. Thank you for being you.

Igor, I cannot measure how special you are to me. I will always thank you for your support, determination and courage to leave everything behind and come with me in this adventure. I always promise you the world, and, although I cannot give it to you right now, we are slowly conquering it together, which is how it is supposed to be. Thank you for your kindness and patience in all these turbulent times. I am grateful for having you in my life.

Also, a thanks to the best friends a person can have. Sana, you are a reference to me. You are

a genius and you have the potential to give an enormous scientific and technical contribution anywhere you chose to work. I hope the people that will work with you will know how to appreciate this. Thank you for the majority of the best moments during the work and for the best travel ever! Ouail, thank you for making the working environment lighter with your kindness and sympathy. You are a brilliant and kindhearted person. It was always a pleasure to work besides you. Bruna, Bianca, Daniel, Sara, May, Thaty, Gaby, Alane, André, Mayã, Andreza, Luiza (and Luna) and Felipe. You are some of my oldest and closest friends, who always encouraged me to follow my dreams and who were always present in my achievements. Only god knows how important you are to me. Thank you very much for everything. I cannot forget Sarah, whose friendship I am proud to keep after all these years. The beginning of my academic career was by your side in the same laboratory and you celebrated with me all my achievements. You are special to me. As always, am looking forward to visit you, take some Açaí and talk about all of our adventures.

To the friends we've made here in Lille, my most sincere thanks. Coming to a new country was really difficult and you've made it a lot easier. You treat us not only as friends, but also as a family and it was really meaningful to us. Talita and Loïc, I do not know what would have happened to us here if we hadn't found you. You were so kind even when you barely knew us. I just can't thank you enough. Furthermore, you've introduced us to a whole group of friends-family that I am proud to be part of. Mizi, Quentin, Aline, Remi, Carol, Stéphane, Liam, Isabel, Karina, Diana, Franciane, Pierre and now the little Constance. You've made our time here to be worthy and precious. Thank you for making Lille a home for us.

Regarding my academic basis, I thank enormously Simon, Philippe and Matthieu, directors and co-directors of this thesis. You trusted me and gave me the opportunity of making this work. More importantly than that, you gave me such support and constructive experiences that I'll never forget. You are now some of the most important basis of my academic career and studies. I'll always be thankful to you.

Similarly, I thank Marcel, the professor who introduced me to the research and taught me some of the most important basis I have. You are a strong reference in my life regarding organisation, honesty and determination. Academically, some of my achievements come from a lot of what I learnt from you.

For the valuable guidance and shared knowledge, I thank all the kindhearted people from Clearsy, specially David, Thierry, Patrick, Etienne and Denis. The experiences I had with you were remarkably enriching. Thank you for taking your time to teach me and improve my work.

I cannot finish this section of my thesis without thanking the people that supported me here at IFSTTAR/Université Gustave Eiffel, specially Nathalie, Joaquin, Corinne, Valérie, Sonia, Olivier and Jorge. You always treated with a great care and helped me to overcome a lot of difficulties. It is not easy to be in a foreign country, specially by the fact that I did not speak french. But, despite of it, you've always been there and made your best to help me with kindness. Nathalie, you are very special and I hope the world will reward you for all your kindness, honesty and determination. Thank you all for everything you have done for me. Corinne, I am always amazed on how you have control of everything you are doing. I cannot imagine IFSTTAR/UGE without you. I am really happy that you were the first person I met when I arrived here and I am really thankful for your kindness and all your efforts on helping me and giving me the best guidance I could receive.

Finally, my thanks goes to the institutes and companies that provided me all the support I needed to accomplish this work: IFSTTAR/Université Gustave Eiffel, Clearsy and SNCF. You gave me the financial, physical and intellectual support that was necessary during the whole journey of this thesis. I am very thankful to you.

Table of contents

General Introduction	1
I Preliminary	9
1 Background	11
1.1 Introduction	12
1.2 Railway Systems and Signalling	12
1.2.1 Safety-critical Aspects in the Railway Field	14
1.2.2 Relay-based and Computer-controlled RIS	17
1.2.3 Case Study	26
1.3 Formal Methods and Mathematical Foundations	30
1.3.1 Formal Specification Methodologies	30
1.3.2 Propositional and First Order Logic	31
1.3.3 Basics of Set Theory and Relations	34
1.3.4 Graph Description Based on Set Theory	38
1.3.5 The B-method	41
2 Formal Specification of Railway Interlocking Systems	47
2.1 Introduction	48
2.2 Timetables-based Approaches	49
2.3 Relay and Computer-based RIS Formal Specification	51
2.3.1 Formalising Relay-based RIS Logic	52
2.3.2 Formal Specification and Implementation of Computer-based RIS	56
2.3.3 Other approaches	59
2.4 Conclusions	61
II Methodology	63
3 Formalisation of Relay-based RIS: A Graph Approach	65
3.1 Introduction	66
3.2 RIS Basic Logical Description	67
3.2.1 Relay Diagrams Basic Structure	67
3.2.2 Electrical Components Structural and Behavioural Formalisation	70
3.2.3 Timed Blocks Impact on the System State Definition	81
3.2.4 Capacitors and their Impact on the System State Definition	86
3.3 Flashing Lights: An Energy Source Variation	94
3.4 Formalisation Support for the System Verification	97
3.4.1 Structural Well-definedness Verification	97
3.4.2 Behavioural Safety Conditions Definition	100
3.5 Case Study Specification and Analysis	102
3.5.1 Structural Formalisation	102

3.5.2 Behavioural Formalisation and Verification	109
3.6 Discussion	113
3.7 Formal Specification Based on the Formalisation	114
4 RIS B Formal Specification: A Diagram-specific Approach	117
4.1 Introduction	118
4.2 Behavioural Specification Based on the System State Space	119
4.2.1 System Variables and State-space Organisation	119
4.2.2 State Evolution Specification	124
4.3 Discussion	136
III Conclusions	141
Conclusions and Perspectives	143
Résumé Étendu en Français	149
Bibliography	159

List of Tables

- 1.1 Some railway signals presented in [Rétiveau 1987]. 15
- 1.2 Representation of electrical components inside relay diagrams. 19
- 1.3 Set notations. 36
- 1.4 Relation, function and sequence notations. 39
- 1.5 B-method notations in unicode and ASCII [Schneider 2001]. 43

- 2.1 Methodologies presented in literature for the relay-based RIS formal specification. 52

- 3.1 Conditions that must be satisfied for the activation of blocks from each type. . 84
- 3.2 Number of allowed connections for each type of component. 99

- 4.1 Components state representation mapping. 122
- 4.2 Representation of buttons and lever contacts states inside the relay diagram. . 123
- 4.3 Inputs values representation mapping. 127
- 4.4 Behavioural and structural definitions that support the relay-based RIS formal specification. 137

- 4.5 Représentation des composants électriques à l’intérieur des diagrammes de relais. 151

List of Figures

1	Diagram presenting the approach proposed in this work for the formal specification of relay-based RIS.	7
1.1	Railway Signalling Levels Scheme.	13
1.2	A monostable relay and its related contacts states.	18
1.3	A bistable relay and its related contact states.	18
1.4	Example of a solution for the ITCS problem used by SNCF.	21
1.5	Example of a circuit that electrifies a light signal.	22
1.6	Indication of the block connections as it is depicted inside a relay diagram. . .	23
1.7	Succession of blocks states according to their types.	23
1.8	Simple succession of a capacitor states.	24
1.9	Representation of a relay-based RIS implemented as a piece of software that communicates with the track-side components through the system inputs (in yellow) and outputs (in green).	26
1.10	Track plan representing the normal and the critical situations of the tracks between the control areas A and C.	27
1.11	Partial relay diagram of the solution for the ITCS problem used by SNCF in the Control Area A.	27
1.12	An example of an electrical circuit and its graph representation.	38
1.13	Directed graph representing the electricity flow inside an electrical circuit. . .	39
1.14	Example of a simple B-machine.	42
2.1	Some relay diagrams modelling styles: (a) Danish [Haxthausen, Kjær, and Le Bliguet 2011], (b) Italian [Cavada et al. 2018], (c) Ladder-like [Van Eijk 1997], (d) French [Sun 2015].	53
3.1	Description of a specific system based on the RIS general structural and behavioural formalisation.	67
3.2	Possible contacts positions in a relay diagram.	70
3.3	Levers states transition example where the small arrows indicate where the current is flowing.	72
3.4	Representation of a contact and its connections.	74
3.5	Impact of the components states over the system state.	76
3.6	A timed activation block (on the left) and a timed deactivation block (on the right).	82
3.7	Different manners to electrify a relay when considering the existence of blocks (the components highlighted in yellow are the ones that are activated).	85
3.8	An example of each capacitor type: Positive-negative, Negative-positive and Bipolar, respectively.	88
3.9	Block adapted to the use of both direct and alternating currents.	95
3.10	ITCS example where components and wires are named.	105
4.1	Representation of the components in the RIS logical formalisation and in the B-method formal specification.	120

4.2	Relay C_CSS_V2 of the ITCS case study representing the states that this component may assume.	121
4.3	Part of the ITCS example showing the separation between the ITCS system and its environment, which are connected by an external relay.	125
4.4	Lever L_ITCS of the ITCS case study representing the states that this component may assume.	126
4.5	Example of a self-powered relay in the ITCS case study.	130
4.6	Étude de cas de l'ITCS.	152

List of Definitions

Structural Definitions

SD1	Relay Diagrams Graph Structure	68
SD2	Components Differentiation	68
SD3	Levers Structure Definition	72
SD4	Monostable Contacts Connections	74
SD5	Bistable Contacts Connections	74
SD6	Bistable Relays Structural Definition	77
SD7	Structural Relation Between Monostable Contacts and Relays	79
SD8	Structural Relation Between Bistable Contacts and Relays	80
SD9	Blocks Differentiation	82
SD10	Blocks Connections Differentiation	83
SD11	Capacitors Structural Differentiation	88
SD12	Capacitors Structural Definition	89
SD13	Capacitors Plates Potential Charge	89

Behavioural Definitions

BD1	Monostable Contacts States Representation	71
BD2	Bistable Contacts States Representation	71
BD3	Buttons States Representation	71
BD4	Levers States Representation	73
BD5	System Disconnections	75
BD6	System Real Connections	75
BD7	Basic Components Electrification Condition	76
BD8	Monostable Relays States Representation	78
BD9	Bistable Relays States Representation	78
BD10	Monostable Relays States Definition	78
BD11	Bistable Relays States Definition	79
BD12	Monostable Contacts States Definition	80
BD13	Bistable Contacts States Definition	80
BD14	Outputs States Representation	81
BD15	Outputs States Definition	81
BD16	Block Passing of Time State Definition	81
BD17	Blocks States Representation	83
BD18	Blocks States Definition	84
BD19	Block-related Electrification Condition	85
BD20	Capacitor Passing of Time State Definition	90
BD21	Capacitors States Representation	91
BD22	Capacitors States Definition	91
BD23	Complete Electrification Condition	92
BD24	Extended Electrification Condition	95

BD25 Extended Blocks Electrification Condition	96
BD26 Outputs Flashing Condition	97

Transformation Directives

TD1 Variables Identification	120
TD2 Variables Typing	121
TD3 Safety Properties Specification	122
TD4 Initialisation Definition	124
TD5 Inputs Identification	125
TD6 Inputs Typing	127
TD7 State Evolution Notation	127
TD8 Output State Succession	128
TD9 Monostable Relays State Succession	129
TD10 Self-powered Monostable Relays State Succession	130
TD11 Bistable Relay State Succession	131
TD12 Timed Activation Block State Succession	132
TD13 Timed Deactivation Block State Succession	132
TD14 Capacitors State Succession	134

List of Publications

The following is a list of publications produced during the course of this PhD research.

Journals

1. **de Almeida Pereira, D. I.**, Perin, M., Bon, P., & Collart-Dutilleul, S. (2019). A framework for the formal specification of relay-based systems based on a b-method graph specification. *International Journal of Computer and Electrical Engineering (IJCEE)*, 11(1), 11-19.
2. Martinez, S., **de Almeida Pereira, D. I.**, Bon, P., Collart-Dutilleul, S. & Perin, M. (2020, July). Towards Safe and Secure Computer Based Railway Interlocking systems. *International Journal of Transport Development and Integration*, 4(3), 218–229.
3. **de Almeida Pereira, D. I.**, Himrane, O., Beugin, J., Bon, P. (2020). From French National Signalling Systems to ERTMS: Considering the Evolution of Track-side Systems. *International Journal of Signal Processing Systems (IJSPPS)*, in press.

Conferences

1. **de 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. In *International Conference on Model and Data Engineering* (pp. 17-29). Springer, Cham.
2. **de Almeida Pereira, D. I.**, Deharbe, D., Perin, M., & Bon, P. (2019, June). B-specification of relay-based railway interlocking systems based on the propositional logic of the system state evolution. In *International Conference on Reliability, Safety, and Security of Railway Systems* (pp. 242-258). Springer, Cham.
3. **de Almeida Pereira, D. I.**, Debbech, S., Perin, M., Bon, P., & Collart-Dutilleul, S. (2019, November). Formal Specification of Environmental Aspects of a Railway Interlocking System Based on a Conceptual Model. In *International Conference on Conceptual Modeling* (pp. 338-351). Springer, Cham.

General Introduction

Context

With the advance of the technology, modern electronic systems are becoming more and more important in the human life as the automation of essential tasks is taking place in the people routines. In some cases, these systems are responsible for such important tasks that their correct functioning is crucial in order to avoid severe repercussions. When the consequences of a failure result in loss of life, significant property or environmental damage or another unacceptable consequence, these systems are considered as safety-critical, which requires more advanced approaches for the verification of their correctness in order to avoid hazardous situations [Knight 2002]. Some examples of these systems are Aircraft Flight Control, Railway, Medical Devices and Nuclear Systems. In this context, the use of modern verification approaches may be the differentiating factor in order to guarantee the safety of these systems.

In the railway context, Railway Interlocking Systems (RIS) are an example of safety-critical systems. The RIS are the part of the railway signalling systems that controls the trains movements in order to prevent hazardous situations like collisions or derailments. With the objective of avoiding the occurrence of several problems like the loss of people lives, injuries, severe environmental damage and economical loss, for instance, the safety of RIS must be guaranteed. Thus, the technologies used by the railway companies for the development of such systems must be able to detect and prevent hazardous situations before their implementation and use.

In general, these systems can be implemented using some different technologies, like relay-based or computer-controlled, this last one being the most recent [Hansen 1998a]. In some cases, the relay-based technology has been used for decades in such a way that the existing interlocking systems are recognised as safe. Nonetheless, despite the historical success of relay-based RIS, computer-based systems are easier to handle and maintain, cheaper and more adaptable to functional requirements changes [Akita et al. 1985]. The use of new technologies is an industrial interest due to their benefits, however, the preservation of the system safety level is a strong requirement in order to replace the existing systems. Thus, the transformation of the existent well succeeded relay-based systems into safety-proved computer-controlled ones can be extremely beneficial in both economical and safety aspects.

In order to prove the safety of a system and support its implementation, Formal Methods may be used. Grounded on a strong mathematical foundation, formal specification methodologies allow the system modelling based on mathematical expressions as a way to define and prove system properties. Currently, there exist several different formal languages documented, each one with a different focus and capable to specify different aspects of the systems. Furthermore, many

formal languages have well-defined refinement methodologies that allow the transformation of the abstract system models into concrete models that may be implemented as software. Some of these languages are supported by tools that automate the process of verification and refinement. Thus, the use of Formal Methods may be the key in order to specify, safety prove and transform the existing relay-based RIS into computer-controlled ones by refinement. The B-method [J.-R. Abrial, Lee, et al. 1991] is one example of a formal language that, together with its supporting tools, allow the system specification, verification, analysis, refinement, implementation and automatic code generation. In fact, the B-method has been already successfully applied in industrial railway projects ([Behm et al. 1999], [Lecomte, Servat, Pouzancre, et al. 2007]).

In this context, the LCHIP (Low Cost High Integrity Platform) Project¹ aims at the implementation of safety-proved computer-based RIS based on the existing relay-based systems used by the French National Railway Company² (SNCF - Société Nationale des Chemins de fer Français). It is a project funded by the Unique Interministerial Fund (FUI) and developed by a consortium coordinated by Clearsy³ and combining the work from different partners, like SNCF and Université Gustave Eiffel⁴. One of the major objectives of this project is to use the B-method as a tool for the specification, safety proving and implementation of the logic behind the existing relay-based RIS in the form of programming code. By following this procedure, it is possible to produce safety-proved computer-based Railway Interlocking Systems whose execution logic is the same of its predecessor technology. Then, these computer-based RIS may run in micro-controllers that can replace the existing relay-based systems as a way to evolve them towards a new technology that is safer, extendable and maintainable.

This doctoral dissertation presents part of the research developed inside the LCHIP Project in the laboratory ESTAS of the Université Gustave Eiffel. It presents a methodology for the specification of relay-based RIS behaviour based on a formalisation of the information contained inside the relay diagrams used by SNCF. The mathematical formalisation of these diagrams as well as their formal specification represent the initial steps towards the implementation of the existing relay-based RIS as computer-based systems as envisaged by the LCHIP Project. In this thesis, the B-method is used as the formal language that supports the specification of these systems. As a result, it is possible to perform proofs regarding the system safety and continue the formal development life-cycle as a way to implement relay-based RIS as computer-based systems.

Problematic

The first built RIS was purely mechanical, than it evolved to use new technologies, becoming electromechanical systems, relay-based systems and, more recently, computer-controlled systems [Hansen 1998a]. Despite the existence of a new technology, relay-based RIS are still used by many railway infrastructure managers, like SNCF. This choice can be explained by the historical success of this technology in addition to the lower complexity and unequivocally defined fault modes [Pasquale et al. 2003]. Although ancient systems have been tested enough to be considered

¹Low Cost High Integrity Platform - <https://www.clearsy.com/en/4260-2/>

²<https://www.sncf.com/>

³<https://www.clearsy.com/>

⁴<https://www.univ-gustave-eiffel.fr/>

safe, their maintenance and the development of new relay-based systems must face a known problem in this type of system: the difficult, time consuming and error prone verification process.

Relay-based Railway Interlocking Systems are generally modelled by electrical circuit drawings named as relay diagrams, which present the structure of the systems based on the electrical connections between the components. This structural model possesses a certain level of formalisation as modelling rules are followed for the system design, although each company defines its own patterns and guidelines. Nevertheless, the lack of a behavioural description makes the safety verification an arduous and error prone process as the system behaviour must be deduced from the relay diagrams. Thus, in order to verify the safety of relay-based RIS, an expert must manually inspect the relay diagrams and draw conclusions about the system safety. *“Due to the high number of diagrams and their mutual correlation, this process is complex, time consuming (and thus expensive) and possibly error prone, which is not satisfactory for a safety critical system”* [Haxthausen, Le Bliguet, and Kjær 2008]. Besides, as each person may have a different interpretation of the system behaviour based on the modelled structure, this process is subject to ambiguity. As a consequence, the models inspection cannot be completely trusted. Moreover, after the manual verification, it is necessary to perform tests in the field. This is an important step in every system as a way to guarantee the correct functioning. Nevertheless, as the system safety cannot be guaranteed, this process can be costly and even risky in some cases.

In such circumstances, the industry needs a more effective approach for the verification of the relay-based Railway Interlocking Systems in order to guarantee their safety. In this context, the European EN50128 guidelines [CENELEC 2011], issued by the European Committee for Electrotechnical Standardisation⁵ (CENELEC), strongly recommends the use of Formal Methods for the specification of systems/components during the development of railway systems. Formal specification methodologies allow the proof of the system safety by modelling its behaviour based on mathematical expressions, which can be used as basis for formal verification processes. So, the use of modern formal specification approaches is not only necessary, but it is also strongly recommended by the railway standards.

Furthermore, as the computer-based technology offers some benefits like a better maintainability and extensibility when compared with the relay-based systems, the railway industry has interest on evolving its systems. Nonetheless, before implementing a new solution, it is imperative to guarantee that the new system safety level is at least equal or better than the precedent technology. This is because the existing relay-based systems are generally already recognised as safe. Moreover, the adaptation of the new system regarding the existing installations must be considered as the industry demands a cost-effective solution.

In conclusion, despite the success of the relay-based technologies, it is a fact that the safety verification of the relay-based RIS can be costly and error prone. As a way to solve this problem and conform to the railway standards, the railway industry needs to be adapted to use modern formal verification approaches, which is able to give a guarantee of the system safety based on their mathematical foundations. Due to the benefits of the use of computer-based technologies, the industry has interest on using this type of technology. Nonetheless, it is important to take precautions in order to maintain the system safety and make this system evolution cost-effective. In this context, the creation of an approach capable of formally verifying the relay-based RIS

⁵<https://www.cenelec.eu/>

and promoting their evolution to computer-based systems maintaining their execution logic can be extremely beneficial to the industry as a way to guarantee the system safety and maintain their previous functioning logic.

Motivations and Objectives

In this present thesis, there is a strong interest on analysing and improving the railway systems as a way to guarantee their safety. The problems faced by the industry regarding the relay-based RIS safety proof require a study about the context of application, the techniques and tools that may be used as well as the experiences of other works in this field. This thesis is also a conciliation between the Railway and Formal Methods areas, which demands a special attention in order to integrate the knowledge, experiences and tools of both fields. Relay-based systems have their own logic and functioning. As a result, the process of formally specifying these systems has a tendency to be a challenging task. A careful analysis of these systems is imperative in order to support their formal specification.

The existence of modern technologies compared to the ones used in the relay-based systems and the industrial interest on evolving their systems are some of the motivations of this work. This is because these technologies can introduce new benefits that were not supported in the preceding systems. Formal specification methodologies has gaining space in the railway field and they have proved their effectiveness by the successful documented experiences about their use. In consonance, the European Committee for Electrotechnical Standardisation recognised the importance of the Formal Methods in this area and strongly recommended its use in the last versions of the norm EN50128 [CENELEC 2011]. All these facts only reinforce the interest on using this technology in our relay-based context.

The formal specification of the existing relay-based Railway Interlocking Systems can support their formal verification as a way to guarantee their safety. Furthermore, the use of a formal specification language that supports a formal development process may allow the implementation of the relay-based RIS as safety-proved computer-based systems. The creation of an approach to support this formal specification is the most fundamental aim of this present thesis, which can be resumed in the following research objective:

RO1: *How to formally specify the existing relay-based RIS as a way to be able to verify and implement these systems by refinement?*

However, it is important to consider that there is a communication problem between the Railway and Formal Methods experts as they may occasionally do not share the same knowledge. Generally, railway experts have a small background on working with formal specification. Similarly, Formal Methods experts typically have none or few experiences on the railway field. In this context, a solution presented for the formal specification of relay-based RIS must be understandable for experts of both fields. Based on these considerations, it is possible to formulate a second research objective as:

RO2: *How to formalise the relay-based RIS in a manner that it can be comprehensible to the different experts involved?*

In the French context, which is the focus of this thesis, relay diagrams are used as models that guide the system implementation. As these models are the most important documentation of these systems, they can also be used to guide an approach for the relay-based RIS formal specification. However, one must take into consideration that each railway company uses a different set of electrical components and different design rules for drawing the relay diagrams. In this situation, it is important to provide a solution that can be extended in order to conform to other contexts and design rules. Thus, it is possible to formulate the third research objective as:

RO3: *How to create a formal model of relay-based RIS that can be extended as a way to support different contexts?*

This latter research question is related to the consideration of the railway system context. However, one must also acknowledge that the formal specification language chosen to specify such systems may impact on the verification that can be performed. This is because every language focuses on different aspects of the system and is able to verify different properties. The literature regarding the use of formal specification languages presents many successful examples that can be adopted, so it is desirable that the RIS formal specification approach can be adapted in order to allow the specification of these systems in different formal languages. In this context, it is possible to define a last research objective:

RO4: *How to define an approach for the formal specification of relay-based RIS that can be adapted to use different formal specification languages?*

As presented in the research objectives, this thesis aims not only at the formal specification of the relay-based Railway Interlocking Systems, but also the development of an approach that can be used in many contexts and that can be adapted for the verification of different aspects of the system. Furthermore, we consider it important the creation of a model that can be understood by different experts involved. The next section presents the main propositions of this work in order to answer these research objectives.

Contributions

After an extensive analysis of the use of Formal Methods for the specification of relay-based Railway Interlocking Systems in literature, it is possible to observe that there are not many works with the objective of formalising these systems. The contributions in this field are mostly focused on the computer-based RIS formal specification and implementation based on a higher level of abstraction of the interlocking procedures. Besides, the existing approaches are generally devoted to the specification of these systems from a specific context and in a specific formal language.

In order to formally specify the existing relay-based RIS, this work proposes a complete structural and behavioural analysis of the relay diagrams used by the SNCF. In this analysis, the structural design and the behavioural logic are studied in order to define the relations between the components and produce a more formal model which can be then used as basis for the system formal specification. By following this approach, it is possible to maintain the system execution logic. Furthermore, it allows the definition and verification of safety properties as a

way to guarantee the system safety.

The formal specification approach chosen in order to specify these systems in this thesis is B-method, due to its successful history in the railway field, strong mathematical background, support to a complete formal development process and the existence of supporting tools for the implementation, analysis, automatic verification and implementation of the systems. Based on the formalisation of the information contained inside the relay diagrams, this work proposes an approach for the adaptation of the formalised logic in order to conform to the B-method syntax. Once formally specified, it is possible to benefit from the advantages of this language in order to verify and implement the relay-based RIS as safety-proved computer-controlled systems, answering the Research Objective 1.

With the objective of formalising the relay-based RIS in a manner that it can be comprehensible to the experts involved, this work proposes the analysis of the relay diagrams based on basic mathematical foundations that are generally studied and understood in all the exact and technological sciences. So, it is proposed to use a graph in order to model the electrical circuit network and the application of First Order Logic and Set Theory as a way to define the structural and behavioural relations between the electrical components. This analysis and formalisation of the systems allow a manual formal verification of the structure well-definedness and the behaviour safety. Furthermore, this analysis proposes a model that can act as a middle course between the structural relay diagrams and the behavioural formal specification, providing a common understanding of the system for the experts of both areas and answering the Research Objective 2.

The use of basic mathematical foundations also provides an extendable and adaptable model that can be adjusted to many different railway contexts, which is our Research Objective 3. The formalised model does not impose limits to the components that may be specified and the logic used allows the specification of the behaviour of a great variety of electrical components. Besides, all the components found in literature are used in the SNCF context. As a consequence, this work presents a non-exhaustive list of components structural and behavioural formalisation that can be used for the system formalisation in many different contexts.

Another benefit of modelling the system with basic mathematical foundations is the possibility of adapting this model to conform with many different formal languages. This is because Set Theory and First Order Logic are some of the most basic foundations of many formal specification languages. So, the expressions used in the relay diagrams formalisation model can be adjusted to the syntax of languages like B-method, for instance. By proposing an approach that can support the formal specification of the relay-based RIS in many different formal specification languages, this work also answers the Research Objective 4.

So, the main contributions of this thesis is the analysis and formalisation of the information contained inside a relay diagram using strong mathematical foundations. This formalisation generates a model that can be adapted and extended in order to conform to different railway contexts. Besides, it can be used to support the formal specification of these relay-based systems in many different formal specification languages due to the common mathematical background. In this work it is also proposed an adaptation approach in order to formally specify the SNCF relay-based RIS in the B-method, which is a method that can support the automatic safety verification and the use of a formal development process for the generation of computer-based

systems. Figure 1 depicts a diagram that presents the approach proposed in this work for the formal specification of the relay-based RIS. The solid lines represent the approaches that are detailed in this thesis, while the dashed lines demonstrate the alternative processes that are supported by these approaches.

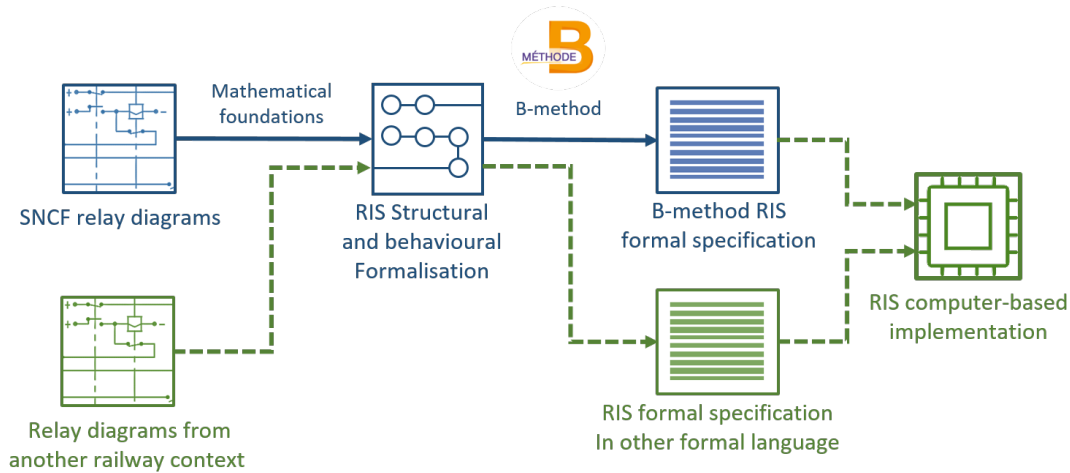


Figure 1 – Diagram presenting the approach proposed in this work for the formal specification of relay-based RIS.

Outline

This manuscript is divided into five main chapters and a general conclusion that are distributed into three parts. The first part of this thesis is devoted to the preliminaries, presenting the background of our work and the state of the art in the literature regarding the formal specification of Railway Interlocking Systems. Then, the second part of this work is devoted to the methodology construction, where it is detailed an approach for the formalisation of the existing relay-based RIS based on mathematical foundations. Then, as presented in this same part, this formalisation may be adapted for the formal specification of these systems in the B formal language. The last part of this thesis is the conclusion, providing a summary of what has been presented and discussing some new research opportunities that result from this work.

In Chapter 1 it is presented the background, i.e., all knowledge that grounds the approaches presented in this thesis. In this chapter, we detail all the information necessary in order to understand the formalism and notations used in the next chapters. It begins by contextualising the railway systems and signalling as it presents the characteristics and safety aspects of these systems. Furthermore, the different types of Railway Interlocking Systems are discussed and detailed and a real industrial case study is presented. Then, this chapter introduces the formal specification methodologies and the mathematical foundations that are used in our approach for the relay-based RIS formalisation. These mathematical foundations are: Propositional and First Order Logic, Set Theory, Relations, and a graph description based on Set Theory. In fact, some of these foundations are also the basis for the B-method syntax, which is also discussed at the end of this chapter. The use of this language in industrial and academic works is also discussed.

Chapter 2 is devoted to the state of the art of this work, i.e., it discusses the use of formal specification methodologies for the analysis, verification and implementation of Railway Interlocking Systems as presented in the literature. This chapter is divided according to the system level of abstraction and the objectives of each presented work. Regarding the level of abstraction, it analyses the use of Formal Methods for the specification of systems focused on the dispatchers and interlocking levels. Then, concerning the latter level, this chapter discusses the approaches that propose the formal specification of relay-based RIS, the ones that propose the implementation of interlocking systems with computer-based technologies and other unusual approaches that contain some similarities to the objectives of this present thesis. Thus, it focuses in positioning our work regarding to what has already been presented in the literature, by discussing how the existing solutions cannot solve our problematic, but are still inspiration to our proposed solutions.

The first chapter of Part II is Chapter 3. In order to be able to formally specify the structure and behaviour of relay-based RIS, this chapter presents a mathematical description of the information contained inside relay diagrams. Based on First Order Logic and Set Theory, it is presented how a graph structure may be used in order to represent the relations between the RIS electrical components. Then, the state of each component is represented and defined in relation to the state of the others. This state definition allows the description of the system behaviour based on the specific behaviour of each component. Furthermore, the impact of the time constraints demanded by some determined components over the system general state is also discussed. As this formalisation is based on mathematical notations, it can already be used as a way to prove determined structural well-definedness and behavioural safety aspects, as it is detailed in this chapter. Then, a case study is analysed according to the generated formalisation model and the results are presented.

The next chapter (Chapter 4) details how the relay-based Railway Interlocking Systems may be formally specified from the relay diagrams by using the behavioural formalisation as a middle course. The B-method is used as formal language and methodology due to its success in the railway field as well as its support for First Order Logic and Set Theory. In this specification approach, the behavioural formalisation described in the previous chapter may be adapted in order to conform to the abstractions provided by the B-method as a way to create a mathematical description that can be verified by the supporting tools. Furthermore, the mathematical description is adapted in order to use the state evolution support given by the formal language as a way to prove the system safety in the complete system state-space. A case study is specified and the results of the formal verification is presented and discussed.

The last part of this thesis (Part III) concludes this manuscript by presenting a summary of the conducted work and the obtained results. Moreover, as this work creates new research opportunities, many future works are presented, like the use of the behavioural formalisation for the specification of relay-based RIS with the use of other different formal languages, or the creation of a specific refinement methodology for the implementation of these systems as computer-based systems with well-defined inputs and outputs. All these research opportunities are detailed and discussed in this last part, concluding that a thesis work is only a small and important step towards multiple research directions.

Part I

Preliminary

Chapter 1

Background

Contents

1.1 Introduction	12
1.2 Railway Systems and Signalling	12
1.2.1 Safety-critical Aspects in the Railway Field	14
1.2.2 Relay-based and Computer-controlled RIS	17
1.2.3 Case Study	26
1.3 Formal Methods and Mathematical Foundations	30
1.3.1 Formal Specification Methodologies	30
1.3.2 Propositional and First Order Logic	31
1.3.3 Basics of Set Theory and Relations	34
1.3.4 Graph Description Based on Set Theory	38
1.3.5 The B-method	41

1.1 Introduction

The railway domain contains several examples of critical systems, whose failures may cause severe consequences like the loss of people lives. The Railway Interlocking Systems (RIS) are one of these examples. As part of the signalling systems, the RIS are responsible for controlling the trains movements in a safe manner in order to avoid hazards. These systems can be implemented by using many different technologies, the computer-controlled systems being the most recent and industrially beneficial.

However, many companies still use relay-based technologies in order to implement the Railway Interlocking Systems. This can be explained by their historical use and the safety provided by this technology regarding possible dysfunctional problems. The transformation of the existing relay-based RIS into computer-based RIS are under the interest of the industry as a way to maintain the system operation with the same or even improved safety level. The use of formal specification methodologies may be the key in order to produce safety-proved computer-based RIS, since their mathematical background may support the formal specification, analysis and verification of the relay-based systems as well as their implementation through refinement.

Nonetheless, before presenting the approaches for the formal specification and implementation of these systems, it is important to understand the role of these Railway Interlocking Systems in the Railway Systems as well as their safety-critical aspects. Furthermore, one must comprehend the differences between the RIS abstract levels and the different technologies that can be used for their implementation.

In this chapter all the details regarding the Railway Interlocking Systems are discussed, which is necessary for understanding the work presented in this thesis. A case study of a real example provided by the French National Railway Company (SNCF) is also detailed as a way to demonstrate the importance of the safety guarantee in these systems.

Then, this chapter provides a discussion about the formal specification methodologies and their use. As a way to support the mathematical formalisation and specification of the RIS, all the mathematical foundations necessary in order to ground our methodology are detailed. These foundations are also the basis of many formal specification languages, like B, for instance. A discussion about the B syntax, supporting tools and successful academical and industrial use concludes this chapter.

All the background information presented in this chapter are essential in order to fully understand the work presented in this thesis. The industrial case study discussed here is used as an example throughout the whole manuscript. Furthermore, the discussion about the safety aspects of these railway systems and the examples of successful use of the B-method in railway industry reinforce the need of the use of formal methodologies for the development of railway systems.

1.2 Railway Systems and Signalling

The railway means of transportation was the first to have mass mechanised movement. After its creation, its velocity, supporting weight and length has constantly increased. According to [Theeg 2017], all railway systems may be identified by two features:

- The train path is determined by the mechanical guidance system comprised by wheels, rails and turnouts;
- The train may move at high speeds in a way that its wheels poor braking response may require a breaking distance longer than what is visible by the driver, so precautions must be taken in order to safely control the train movements.

On its most basic structure, a railway system is composed by steel wheels, rails (tracks) and turnouts, being this last one the way how trains may change their direction. The steel material allows the system to withstand heavy transits, but its low adhesion coefficient impacts negatively on the breaking capabilities. In order to make the regulation of traffic and the prevention of accidents, Railway Signalling Systems are responsible for detecting data like the trains positions and track availability, process them and control the trains movements and other track components.

Railway Signalling Systems are subdivided into three levels: the Element, Interlocking and Operation Control levels, as presented in Figure 1.1. The Element Control Level is the interaction between the system and field elements like train detectors, turnouts and signals. This is the part of the system responsible for controlling and monitoring the track electrical and mechanical components. The Interlocking Level is the part of the system responsible for processing the data and responding accordingly to safety aspects as a way to avoid dangerous situations. Then, the Operation Control Level is the interface between the system and the signaller, i.e., the person that induces the train movement. Thus the safety of the system depends on the Interlocking Level capabilities of processing the data and sending the correct information to both Element and Operation Control levels.

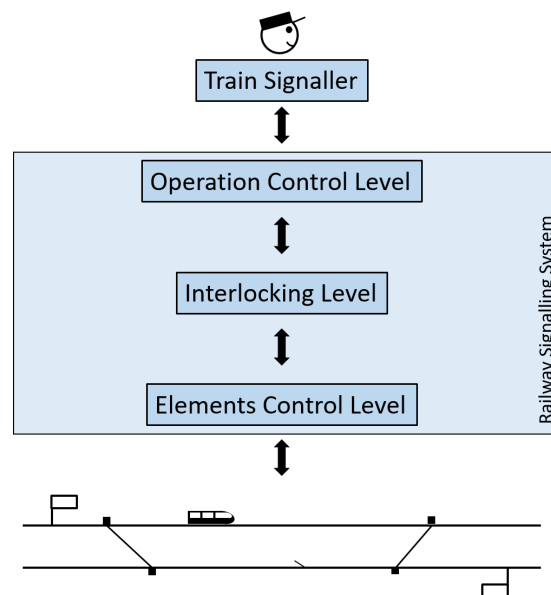


Figure 1.1 – Railway Signalling Levels Scheme.

Above the Operation Control Level, there is the train signaller, which is the responsible for directly inducing the train movement. The signaller decisions are made based on timetables, signalling rules and the current situation. The train traffic management schedules are produced

in the form of timetables by the dispatchers, which have the objective of organising a well-thought train schedule and communicate with signallers any unexpected situation. The Dispatchers Level is right above the Signaller Level in the Railway Signalling level hierarchy.

In order to guarantee the safety of these Railway Interlocking Systems, the Interlocking and the Dispatchers levels must be analysed. In the Interlocking Level it is necessary to guarantee that the system is indeed executing accordingly to safety aspects and avoiding any hazardous situations, which requires a deep analysis of the logic behind these systems. In the Dispatchers Level, one must assure that the timetables do not cause a dangerous situation by avoiding any proximity between trains during their route. Although these analysis have the same objective of guaranteeing the system safety, they are made in completely different manners that require different approaches. While the verification of interlocking systems is made regarding a specific local situation, the verification of timetables requires the analysis of several train routing in order to guarantee that they do not share the same track point. Both works are extremely important in order to guarantee the system safety and many methodologies and studies have been made in both areas. This thesis focuses on the analysis and verification of the systems in the Interlocking Level.

1.2.1 Safety-critical Aspects in the Railway Field

The main concern about safety-critical systems is with the consequences of failure [Knight 2002]. In this case, a failure may be defined as an external incorrect behaviour according to the system requirements and the expected behaviour [Ammann 2016]. When a failure leads to acknowledged unacceptable consequences, the system is determined as safety-critical. Some well known examples of traditional areas where safety-critical systems are applied are medical devices, aircraft flight control, weapons, and nuclear systems [Knight 2002]. In the railway field, the safety-critical nature of railway systems is evident, since a failure may cause severe consequences like the loss of people lives, substantial economical loss and even extensive environmental damage.

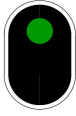



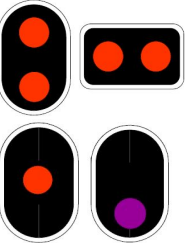
The safety in this context is related to the *"functional safety within the system and protection against hazardous consequences caused by technical failure and unintended human mistakes"* [Theeg 2017]. Given that an error is an incorrect internal state of the system [Ammann 2016], the system safety may be achieved by avoiding the occurrence of errors through a careful inspection before putting the system into operation. Nevertheless, *"a spontaneous (random) failure during operation cannot be prevented. However, dangerous consequences of such a failure can be prevented by the design of the system"* [Theeg 2017].

According to [Schön et al. 2013], the railway operation presents five major problems:

1. Collision between trains that go in the same direction in the same track in different speeds;
2. Collision between trains that have converging routes;
3. Frontal collision between trains that travel in the same track in opposite directions;
4. Collisions at a road level crossing;
5. Derailments, which may be caused by excessive speed, for instance.

Except for derailments, all the other cited problems are related to collisions while the train is under its route. The causes of these accidents are numerous, but the human factor and equipment defect are some of the most significant causes [Liu, Saat, and Barkan 2012]. Regarding derailments, for instance, it is known that at low speed, they are mostly caused by certain track and human factors, like improper train handling, braking operations and improper use of turnouts (points). At higher speeds, these problems are mostly caused by equipment defects [Liu, Saat, and Barkan 2012]. While human erroneous decisions are difficult to predict and control, the correct operation of interlocking systems may guarantee the non-occurrence of some problems when considering that their instructions are well followed by all the related humans. In this case, although the signalling installations can solve the railway operation problems, it does not diminish the importance of regulation, since the obedience to signal indications and exceptional procedures are a matter of regulation [Rétiveau 1987].

Table 1.1 – Some railway signals presented in [Rétiveau 1987].

	<p>Green Signal - Normal operation is authorised, if there are no objections.</p>
	<p>Yellow Signal - It is necessary to be able to stop before the next stop signal.</p>
	<p>Double horizontal yellow Signal - Commands to not exceed the speed of 30km/h when passing over the corresponding turnout.</p>
	<p>Double horizontal flashing yellow Signal - Commands to not exceed the speed of 60km/h when passing over the corresponding turnout.</p>
	<p>Red and purple Signals - Instructs to stop in front of the signal.</p>

In order to indicate the safe procedure for a train in its route, railway signals are used. In the french context, light signals are used as stop, speed limit and direction signs [Rétiveau 1987], as presented in Table 1.1. As the interlocking system is also responsible for controlling the railway turnouts, they must guarantee that a train only passes through them when they are placed and locked, since an unexpected switch movement may cause the derailment of the train that is using it. Together with the signals that control the train speed, a RIS whose instructions are well followed have the responsibility to avoid the occurrence of collisions and derailments in

several different situations. Therefore, the logic of these systems must be safety proved as a way to guarantee the absence of accidents.

In the European context, the electrical engineering standardisation is a responsibility of CENELEC, the European Committee for Electrotechnical Standardisation (Comité Européen de Normalisation Électrotechnique). Some of the most important European Norms (EN) regarding the railway systems development and operation are the EN 50126 [CENELEC 2017a][CENELEC 2017b], EN 50128 [CENELEC 2011] and EN 50129 [CENELEC 2018].

Regarding the safety, EN 50126-1 [CENELEC 2017a] provides a Safety Management Process for the Railway Systems development. This process is supported by the guidance and methods presented in the EN 50126-2[CENELEC 2017b]. The approach defined in EN 50126 is consistent with the application of quality management requirements defined in the ISO 9001 [ISO 2015]. This norm also defines several terms, like:

- Accident – *“unintended event or series of events that results in death, injury, loss of a system or service, or environmental damage”*;
- Error – *“discrepancy between a computed, observed or measured value or condition and the true, specified or theoretically correct value or condition”*;
- Failure – *“loss of ability to perform as required”*;
- Hazard – *“condition that could lead to an accident”*;
- Reliability – *“ability to perform as required, without failure, for a given time interval, under given conditions”*;
- Risk – *“combination of expected frequency of loss and the expected degree of severity of that loss”*;
- Safety – *“freedom from unacceptable risk”*;
- Safe state – *“condition which continues to preserve safety”*;
- System – *“set of interrelated elements considered in a defined context as a whole and separated from their environment”*;
- Subsystem – *“part of a system, which is itself a system”*;
- Verification – *“confirmation, through the provision of objective evidence, that specified requirements have been fulfilled”*.

While EN 50126 addresses system issues in a widest scale, EN 50128 concentrates on the methods for the development of software that complies with the safety demands. In this context, Software is defined as *“intellectual creation comprising the programs, procedures, rules, data and any associated documentation pertaining to the operation of a system”*. This norm also defines five software Safety Integrity Levels (SIL), being SIL0 the lowest and SIL4 the highest one, measuring the risk resulting from software failure. Furthermore, the EN 50128 gives recommendations of techniques and measures according to each Safety Integrity Level. For instance, the use of Formal Methods of specification based on a mathematical approach is recommended for systems SIL1 and SIL2, but it is highly recommended for systems SIL3 and SIL4.

Regarding the electronic part of the signalling system, the safety-related acceptance requirements of electronic systems are defined in the EN 50129. During the execution of a signalling system, one must consider that problems caused by electrical components defects are hard to predict and prevent. These problems can be mitigated by the use of ancient relay-based Railway Interlocking Systems, which have lower complexity and unequivocally defined fault modes [Pasquale et al. 2003]. Nevertheless, these systems are difficult to model, safety proving and maintain. Thus, they are being replaced by computer-controlled systems, which are easier to handle and maintain, cheaper and more flexible to extend functions [Akita et al. 1985]. Each of these technologies have their advantages and disadvantages, which are discussed in the next section.

1.2.2 Relay-based and Computer-controlled RIS

In the beginnings of the railway operations, all the interlocking procedures were made by humans, which had the responsibility of manually interacting with the field elements [Theeg 2017]. This "interlocking" procedure is not a real interlocking, since no technical locks are provided. For safety reasons, this procedure was widely replaced by mechanical systems. In fact, the first Railway Interlocking System was purely mechanical. Then, as electricity became common, the mechanical systems evolved to electromechanical relay-based systems. More recently, computer-based technology is replacing the electrical systems [Hansen 1998a]. Nowadays, many railway infrastructure managers are replacing the existing relay-based systems by computer-based technologies.

Some of the first steps towards the use of electrical components in the RIS started around 1870, with the development of partial electrical systems beginning around 1900. But it was only between the two world wars that the firsts relay-based RIS were developed and installed in various countries. This type of system is still used in the majority of existing installations [Theeg 2017]. However, With the existence of more advanced technologies, the relay-based systems are being replaced by the computer-based ones (electronics).

Relay-based Systems and Modelling

Relay-based RIS are implemented in the form of electrical circuits whose electrical current flux is controlled by relays. As an electromagnetic component, a relay is composed by a electromagnet (coil) and a movable armature containing one or more electrical contacts. When electrified, the relay coil produces a magnetic field that attracts the armature, changing the contacts positions, which may open or close circuits according to their initial positions. Figure 1.2 depicts the states of a relay R and its related contacts $C1$, $C2$ and $C3$. By controlling the flux of electrical current in other wires, the alteration between the relay states may activate or deactivate other relays, which creates a chain effect until the system reaches a stationary state, i.e., the moment where no component has its state altered.

Relays are divided into two different kinds: Monostable and Bistable. The main difference between them is the impact of the gravity on their states. A monostable relay contains only one electromagnetic coil that pulls the contacts against gravity. In this case, the contacts are physically disposed horizontally so they can fall down when the coil is not energised, as presented in the Figure 1.2. A bistable relay, on the other hand, has two electromagnetic coils that pull

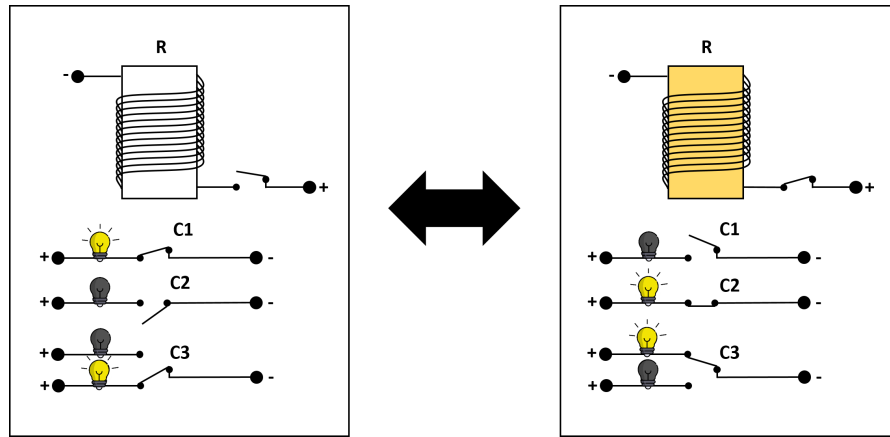


Figure 1.2 – A monostable relay and its related contacts states.

vertically positioned contacts to different sides, as presented in the Figure 1.3. The contacts are attracted to the energised coil. The relay coils positions are typically called as "left" and "right" [Schön et al. 2014]. Furthermore, if both bistable relay coils are activated or deactivated, gravity causes the contacts to maintain their previous states.

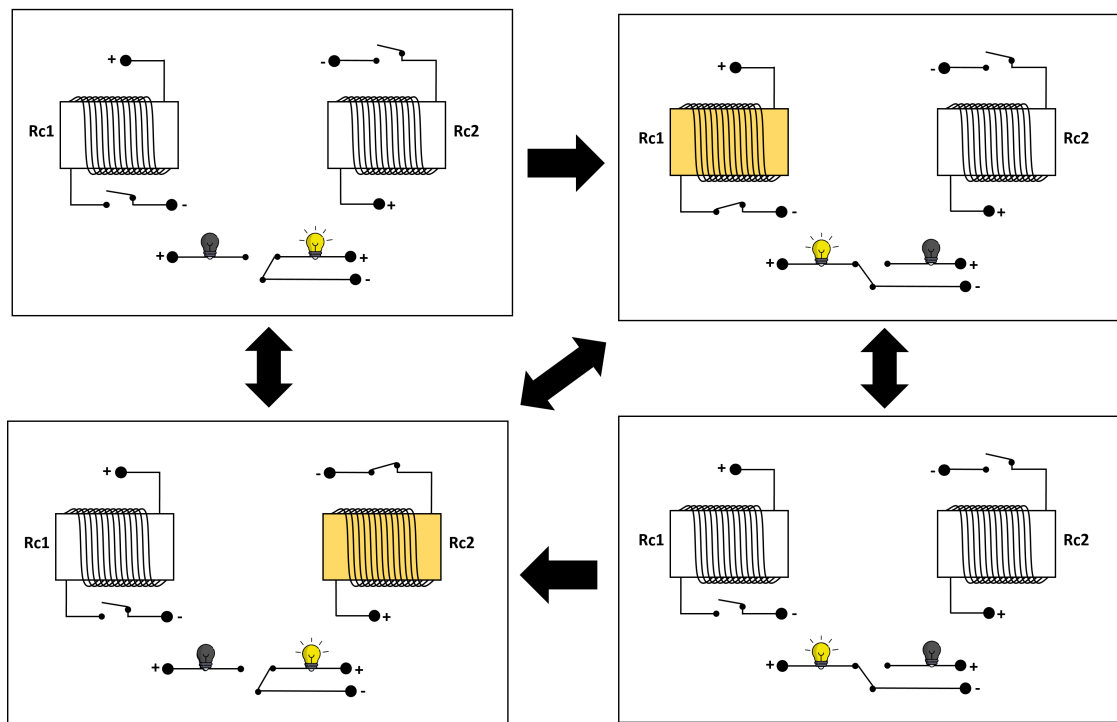


Figure 1.3 – A bistable relay and its related contact states.

The contacts related to monostable relays, the monostable contacts, may be divided into three categories: normally-open, normally-closed and changeover contacts [Schön et al. 2014], which are represented in Figure 1.2 as the contacts *C1*, *C2* and *C3*, respectively. A normally-open contact is open when the relay is deactivated and closed otherwise. On the other hand, a

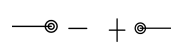

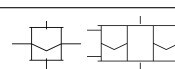
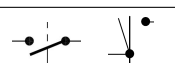
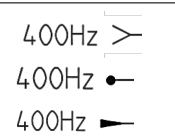
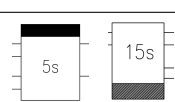
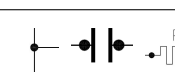
normally-closed contact is closed when the relay is deactivated and open otherwise. In the case where a contact is able to establish a connection independently of the relay state, it is called as a changeover contact, since it is able alternate between two different connections.

The differentiation of contacts types is important because the most stable state of a monostable relay is the deactivated state, i.e., when it is not energised. As the contact falls down by gravity, this position is called as the "safe position" as it is used in order to keep the system safety in case of a component failure. In this case, the system must always give an information that can only leads to a safe state, thus the "down" position of a contact generally leads signals to be red or detectors to indicate a train presence. Besides, relays are made following strict compliance requirements so they can be trusted during a system execution. An example of the dysfunctional safety guarantee given by relays is detailed in the case study presented in Section 1.2.3.

Before their implementation as electrical circuits, relay-based RIS are generally modelled as electrical circuits schemata named relay diagrams. They represent in a graph format how the electrical components are connected by wires. Nonetheless, there is not a unique approach for drawing relay diagrams, as each company has its own set of electrical components and design rules. Although some diagrams may be similar, their representation may have a big impact on how the system behaviour may be understood from the drawings. This thesis focuses on the relay diagrams representations given by the French National Railway Company (SNCF) and their documentation in [Rétiveau 1987] and [Schön et al. 2014].

Besides the relays, many other components may be used in the implementation of relay-based RIS and modelled inside a relay diagram. Table 1.2 presents how some of these electrical components are graphically represented. Each component has its specific behaviour and plays an important part on the complete system execution.

Table 1.2 – Representation of electrical components inside relay diagrams.

	Energy sources.
	A lever and a button, respectively.
	Monostable and bistable relays, respectively.
	A monostable and a bistable contact, respectively.
	Alternating current energy sources.
	Blocks for timed activation and deactivation, respectively.
	A junction, a capacitor and a resistor, respectively.

The energy sources are responsible for providing energy to the system in order to activate

determined components. Generally, a component is electrified when connected to both energy sources poles, negative and positive. Nonetheless, components may also be electrified when connected to other more complex components, like blocks or capacitors, as discussed later. However, even these components require a connection to energy sources in order to function correctly.

It is also possible to define an interface that allows the system user to control the electricity flow inside the wires as well as obtain important information about the system state. This interface may be implemented by the use of buttons and levers as the system inputs, and lights and antennas for the system outputs. A button allows the environment to control the current flow in one single wire, since this component acts like a contact that requires physical force in order to close. A little more complex than buttons, levers allow one to control the flux of electrical current in many wires at the same time. Levers are connected to a set of contacts which always alternate their states together. When the lever state changes by an environmental physical force, all its related contacts alternate their states together, blocking and allowing the current to flow in different wires at the same time.

Monostable relays and bistable relays are represented inside relay diagrams as single and doubled coils, respectively. Each coil has two independent connections to wires. The relation between relays and contacts are represented as a semi-dotted vertical line, as presented in the example of the Figure 1.4. This figure represents a solution for a Temporary Reversed Direction Installation (ITCS - Installations Temporaires de Contre Sens), given by SNCF, discussed with more details in Section 1.2.3.

As a way to improve readability, bistable relays and levers may have specific names for each of their states. These names are generally related to the effect of these components states over the rest of the system. A lever may have the names "on" and "off" for each of its states and a bistable relay may have the states "in service" and "out of service" instead of "left" and "right", for instance.

Relays may also be considered as inputs or outputs of the system in determined situations. When the relay is not presented inside the diagram but its related contacts have influence on the system execution, this relay is considered as an input. In this case, the relay is considered as part of the environment, since it is not controlled by the modelled system. Furthermore, if the relay is presented inside the diagram, but its related contacts are not part of it, this relay is an output, since it impacts the environment through its contacts. Indeed, any relay may be an output, as long as it has at least one contact that is not presented inside the diagram. In this case, the interpretation of the diagram and the knowledge about the system is extremely important in order to define what is indeed an output or not.

One of the possible system outputs are the light signals. Differently from the rest of the system, these outputs are powered with an Alternating Current (AC) of 400Hz. The alternating current is only a variation of the energy sources used in the relay diagrams. As this type of current is only used for signal lights, this work refers to the Direct Current (DC) energy sources as the "energy sources". Otherwise, when alternating current energy sources are used, they are specifically referred as "alternating current energy sources" (or "AC energy sources"). This is because the majority of the system uses DC energy sources, so there is no need to state this every single time. In order to differentiate the AC energy sources, some of them may be modelled

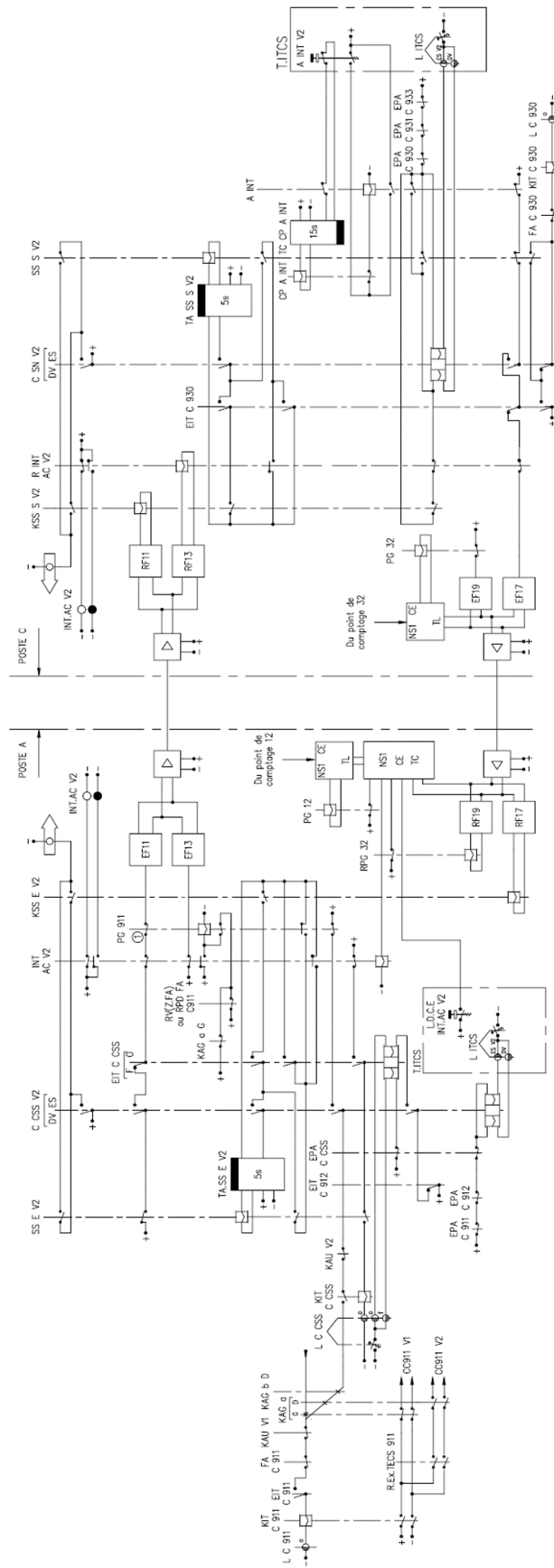


Figure 1.4 – Example of a solution for the ITCS problem used by SNCF.

in the relay diagram with the word "retour" ("return", in english). So, in order for a light to be electrified, it must be connected to a normal and a return AC terminals.

Other important characteristic of the signals is that they can be flashing or fixed. In order for the light to flash, the AC energy source flashes. In this case, in order to indicate that the alternating current of the energy source is flashing, the wire coming out of this component is represented by a dashed line. The Figure 1.5 presents an example of a circuit using the AC energy sources giving the possibility of the light signals to flash or not.

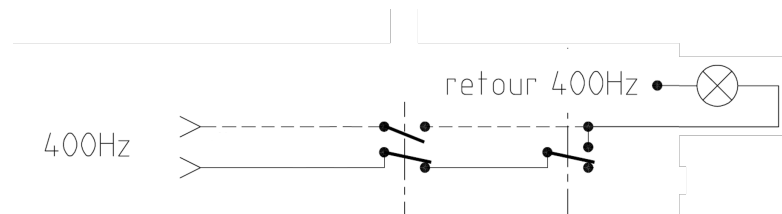


Figure 1.5 – Example of a circuit that electrifies a light signal.

During the execution of relay-based RIS, the existence of timed behaviours may be necessary. As electrical components may physically require some time to reach their determined states, the system safety may be guaranteed by waiting specific times in order to change signals, for instance. In this context, complex structures used in the system implementation are responsible for timely controlling the electricity flow. These structures are abstracted inside relay diagrams as blocks.

A block is generally drawn as a box with many different connection. They represent complex structures that are not explicitly depicted in the diagram. There are many types of blocks with different functions. In this work we consider only the existence of timed activation and deactivation blocks. These components may generally have five or six connections that may be classified as:

- positive energy connection;
- negative energy connection;
- one or two independent connections, which are responsible for energising the block itself when the configuration allows the current to flow;
- two dependent connections, whose electricity flows from and to the block when it is activated.

The block connections are presented in Figure 1.6. Indeed, there is no visual differentiation between the block independent and dependent connections. Although only the dependent connections require the block to be activated in order to produce energy, both of these connections may act as power sources.

Differently from other components, a block is electrified if there is a cycle that begins and finishes in the block independent connections or if there is a connection between an independent connection and an energy source, as presented in Figure 1.7. However, a block may not be activated right after its electrification. In this context, blocks may be divided into two different types:

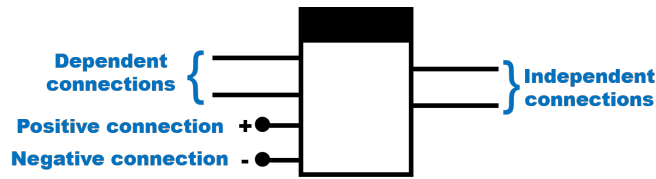


Figure 1.6 – Indication of the block connections as it is depicted inside a relay diagram.

- Blocks with timed activation - represented in the relay diagram with a thicker line on the top, these components activate a certain time after their electrification, but they deactivate right after the energy is cut;
- Blocks with timed deactivation - represented in the relay diagram with a thicker line on the bottom, these components activate right after their electrification, but they deactivate only after a certain time that the energy is cut.

When activated, the block produces energy to the circuit connected to its dependent connections. The complete succession of blocks states is represented in Figure 1.7. In this figure, the components in yellow are activated, while the components in white are deactivated. The first column of this figure depicts the blocks electrification, which activates the timed deactivation blocks. After X seconds, the timed activation blocks also activates (second column). When these components are no longer electrified, as represented in the third column, the timed activation block deactivate. The timed deactivation blocks, on the other hand, only deactivate after X seconds, as depicted in the fourth column.

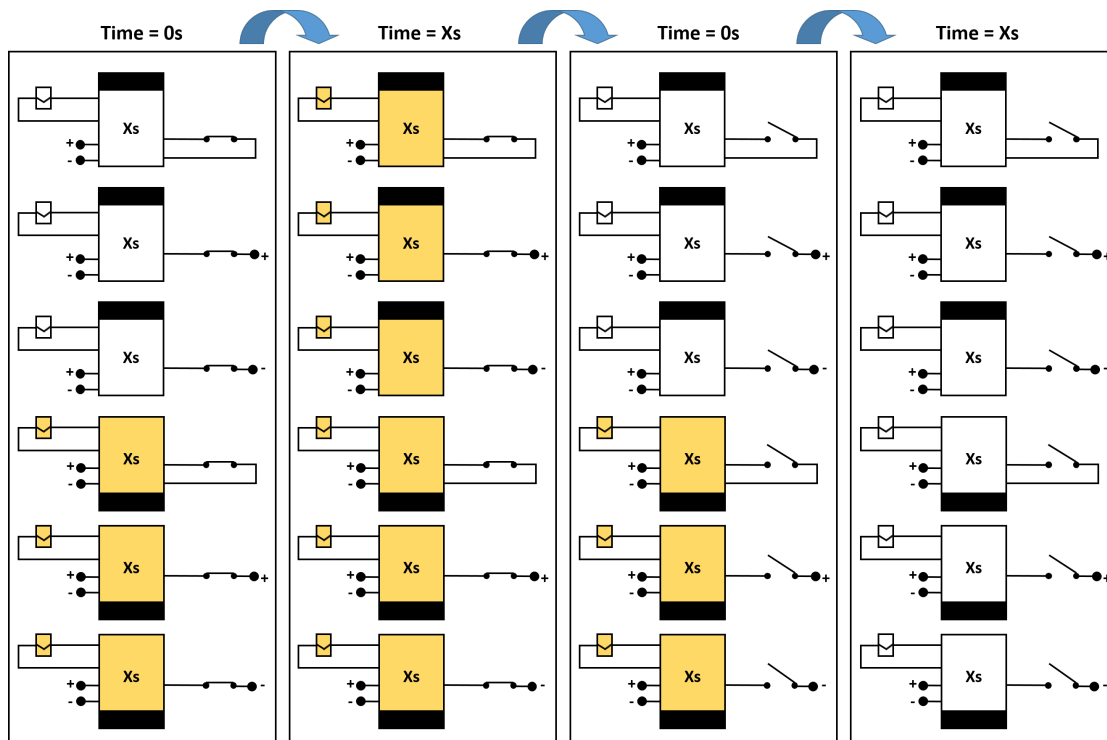


Figure 1.7 – Succession of blocks states according to their types.

An electrical junction allows a component to be connected to many others at the same time. It makes the contact between three or four wires as a way that the electrical current that comes from a wire is able to pass through all the others. In fact, this component has no behaviour, but it is an important support component during the design and implementation of relay-based RIS.

The component with the most complex behaviour is the capacitor. This component is able to store energy after a certain time that it is energised, which can be discharged during a certain time in order to electrify other components. So, its behaviour is time dependent when charging and discharging. Besides, a capacitor is only able to electrify other components if it is completely charged, which makes this component state to be dependent from its previous state. A capacitor consists of two conductive regions (plates) separated by a non conductive material. Thus, this component has the ability of storing a different energy charge in each of its conductive regions according to the energy source pole they are connected to. When each plate of a capacitor is connected to a different energy source pole for a certain time, the plate connected to the positive pole accumulates positive charges while the plate connected to the negative pole accumulate negative ones. A capacitor possible state succession is presented in Figure 1.8.

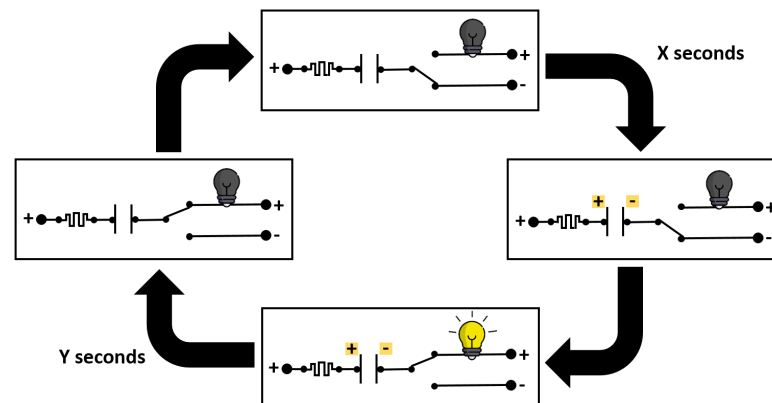


Figure 1.8 – Simple succession of a capacitor states.

As well as junctions, resistors are one type of component that has no behaviour when electrified. On the other hand, these components are used together with capacitors (as presented in Figure 1.8) as the resistor level may control the time for capacitors to charge. They have an structural and behavioural importance, as they must be physically configured in order to fulfil the capacitors time requirements.

Generally, the safety verification of relay diagrams is made by human inspection, which may not be satisfactory for a safety-critical system [Haxthausen, Le Bliguet, and Kjær 2008]. A structural verification aims to guarantee that the electrical circuits are well-defined, i.e., all the components electrical connections follow the electrical circuits design rules. The behavioural verification, however, is the interpretation of these diagrams as a way to investigate if the inputs generate the expected outputs. Due to the complexity of these systems, a manual inspection has a high tendency to be time consuming and error prone, besides, the diagrams may require high level of expertise in order to be correctly understood. Despite the historical success of this technology and its well defined fault modes, relay-based RIS are being replaced by a more intelligent and modern technology: computer-based RIS.

The New RIS Generation of Computer-controlled Systems

Firstly applied in the 1980s, the electronic RIS have been continuously developed in a way that many different versions of systems have been created by different manufacturers and applied in different countries [Theeg 2017]. As computer-based systems, the interlocking functions are programmed in a software format. In fact, these electronic systems have some risky characteristics. One must consider that the low voltage used makes these systems highly sensitive to external influences and that their failures are difficult to predict. Furthermore, the hardware of these systems are always evolving and they are complex, in a way that it is hard to prevent manufacturing errors and guarantee the well functioning of the system.

However, as these systems have a tendency to be cheaper and flexible, many different approaches may be applied in order to overcome the unfavourable situations. For instance, hardware safety redundancy may be used as a way to execute the logic in different pieces of hardware and compare the results as a way to produce a final decision. This approach allows the exclusion of spontaneous errors. Furthermore, hardware availability and redundancy may be used in order to allow the system to maintain its operation when some piece of hardware fails. In this approach, the system maintains extra components that may be automatically used in case of failure as a way to guarantee the system operation.

The interface of a computer-based RIS, i.e., its Operation Control Level, is generally provided as a remote control system, which sometimes is not even considered as part of the interlocking system. However, the Interlocking and Element Control Levels are still part of the interlocking system and they may be implemented with the use of many different technologies. As computer systems are always evolving, new methodologies for increasing the RIS safety are continuously being studied. A computer-based RIS may also make diagnosis in order to check the correct operation of components in all system levels. This part of the system may sometimes be allocated in a different hardware block [Theeg 2017].

As the interlocking logic is concentrated in the form of software, the system that used to be implemented as a big web of cables and electric components may be reduced to small electronic boards that execute software, as depicted in Figure 1.9. The communication with track-side components as sensors, signals and turnouts can be maintained and all the system logic may be centralised and easily processed in order to give multiple useful information to operators.

Furthermore, as part of the Software Engineering practices, formal approaches for software development may be applied in order to meet safety requirements. The use of formal specification methodologies in the railway field is indeed strongly recommended by the railway norms [CENELEC 2011]. This can be explained by the Formal Methods strong mathematical background and supporting tools that allow the automatic safety analysis and proof. Some formal languages have already been used in industry with successful results. One of the examples is the B-method, which is considered as one of the strongest approaches for the specification of railway systems [Fantechi, Fokkink, and Morzenti 2013] and which has already been successfully used for the analysis and development of software in many different projects, like METEOR [Behm et al. 1999], COPPILOT [Lecomte, Servat, Pouzancre, et al. 2007] and SACEM [Guiho and Hennebert 1990]. Other examples of languages used in this context are Petri Nets [Peterson 1977], CSP [Schneider 2000] and Z [Spivey and J. Abrial 1992], used in works like [Sun, Collart-dutilleul, and Bon 2015], [Winter 2002] and [N. A. Zafar 2006], respectively, as discussed in the

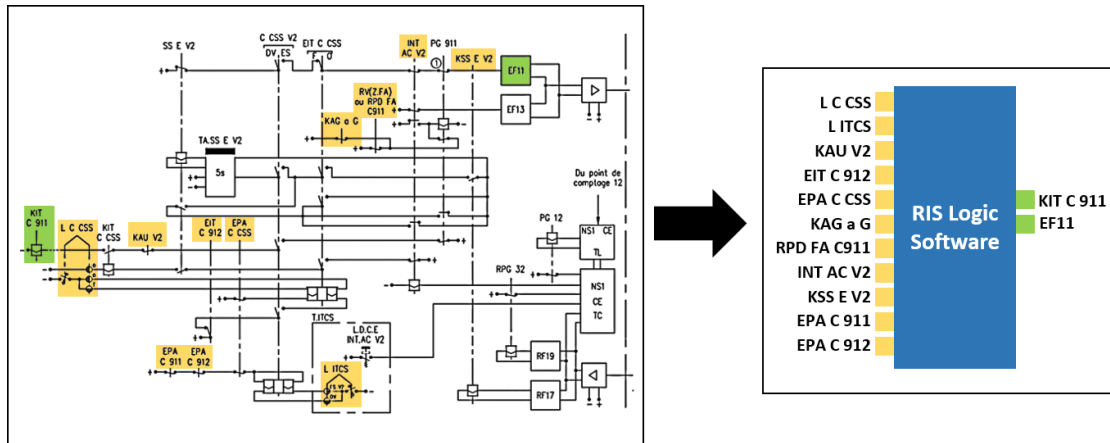


Figure 1.9 – Representation of a relay-based RIS implemented as a piece of software that communicates with the track-side components through the system inputs (in yellow) and outputs (in green).

Chapter 2.

1.2.3 Case Study

A well known example of a Railway Interlocking System safety application is the control of the railway signals during a Temporary Reversed Direction Installation (ITCS - Installation Temporaire de Contre Sens). In a normal situation of a pair of tracks, each track is used for the trains to go in one different direction. However, some determined circumstances may cause a track to be blocked, either because of repairs, maintenance or even accidents. In order to maintain the flux of trains, a special approach must be used. The ITCS system uses two turnouts in order to allow the trains that should pass through the blocked tracks to temporarily use the track in the wrong direction in a safe manner.

Relay-based RIS are divided into control areas. Each control area contains a small portion of the system, which is responsible for controlling the signals and turnouts inside the limits of this area. The communication between the control areas can be performed by specific components, like antennas, for instance. Figure 1.10 presents the track plan of the ITCS example between the control areas A and C, depicting the normal and the critical situations. In a normal situation, the trains are able to transit between the areas in tracks reserved for each specific direction. Then, due to a problem in one of the tracks, the train may continue its route by passing through the other track in a temporary installation, going in the opposite direction. However, when a train that comes from a Control Area A changes to the other track, it may cause a collision with a train that comes from the Control Area C. In order to avoid this collision, the interlocking system must be able to detect the train presence and control the signals according to safety principles.

In order to solve this problem, many different solutions may be given. One of the solutions used by SNCF and installed in the Control Point A is partially presented in Figure 1.11. This is the part of the diagram presented in Figure 1.4 responsible for controlling the signal in the Control Area A. Besides, according to the context in this control point, this system also indicates the track availability to the Control Area C through the output *EF11*. Each electrical component

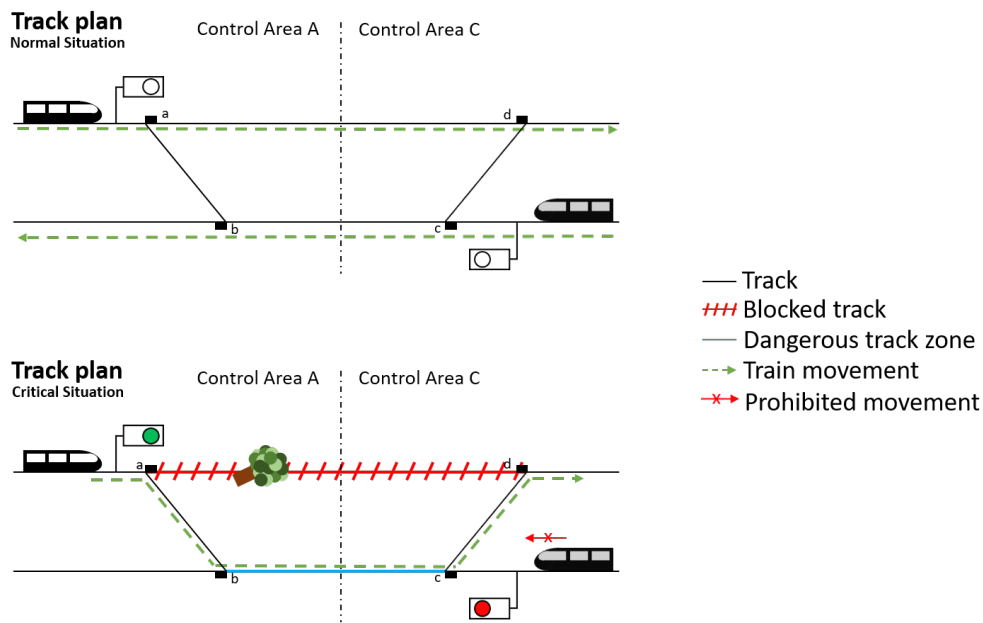


Figure 1.10 – Track plan representing the normal and the critical situations of the tracks between the control areas A and C.

in this diagram has an importance in order to guarantee the system safety.

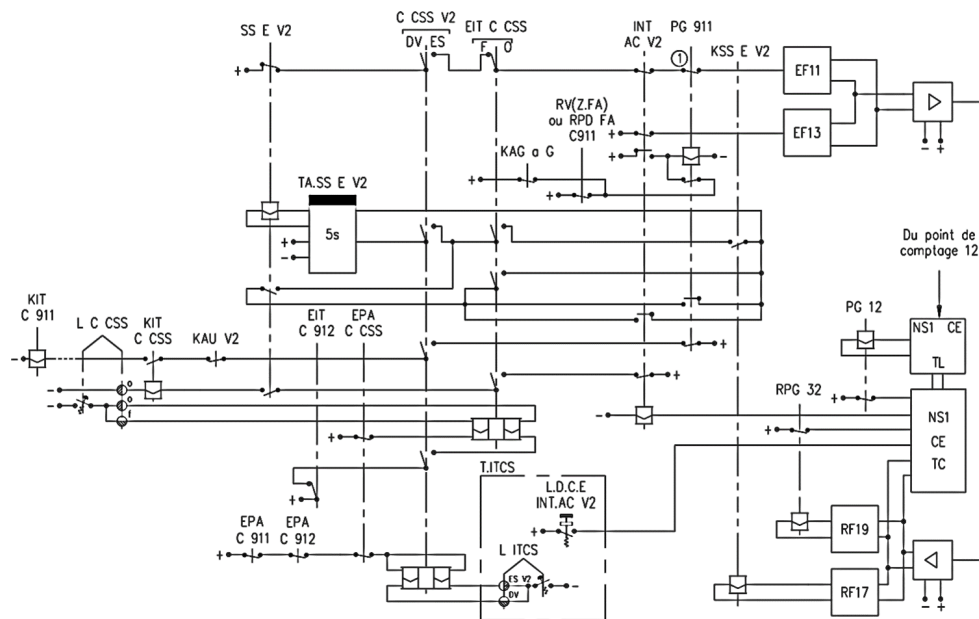


Figure 1.11 – Partial relay diagram of the solution for the ITCS problem used by SNCF in the Control Area A.

This case study begins when a problem in the track is detected. The lever *L ITCS* is responsible for indicating if the itinerary of the train that comes from Control Area A must pass through the opposite direction. In this case, this lever may assume the states *DV* (two-way route - Double Voie) or *ES* (in service - En Service). When this component is in service, the system must provide

a safe mechanism for a train that comes from the Control Area A to pass through the opposite direction track for a moment. This lever state induces the bistable relay *C CSS V2* to go likewise to the *ES* state. This component is responsible for the control of the opposite direction route.

Once the exceptional (critical) situation is set, the system waits for the permission for the train that comes from the Control Area A to continue its route in the opposite direction. This permission is given by the Lever *L C CSS*. When the state of this lever is changed from *F* (closed - Fermé) to *O* (opened - Ouvert), the procedure in order to allow the train to change tracks begins. The objective of the system at this moment is to block the trains that may come in the normal direction so a train may pass in the opposite direction. With the critical situation set and the permission given, the control command *EIT C CSS*, a bistable relay, is also set to the *O* state. This command already deactivates the output *EF11*, indicating to the Control Area C that an ITCS procedure is beginning.

Then, the system waits for a permission from the Control Area C in order to continue the procedure. This permission is given by the component *KSS E V2*. The activation of this component provokes the electrification of the block *TA.SS E V2* and its activation after five seconds (the time indicated inside the block). This time is a security measure in order to make the train wait five seconds more before entering in the dangerous zone (the shared portion of the tracks where a collision may occur). In this time, any sensor that may have been delayed for any reason may indicate any possible presence in the tracks. After five seconds, the block activates, provoking the activation of the safety relay *SS E V2*. If nothing more blocks the entrance of the train, the activation of the relay *SS E V2* leads to the activation of the relay *KIT C CSS*, which is the final command in order to activate the *KIT C 911*. Then, the train is able to safely enter in the dangerous zone.

The activation of the component *KIT C 911* could be prevented by other components in two main cases. The first case occurs when the axle counter detects that part of a train is still in the dangerous zone. This component is responsible for guaranteeing that the same amount of wheels that entered in a determined portion of the track is the same amount that left, i.e., no rolling stock is missing. In this ITCS example when the component *INT AC V2* is deactivated, it indicates that part of the train is still in the dangerous zone, which blocks the activation of the component *KIT C CSS*. The second case occurs if the pedal in the dangerous zone detects the presence of a train (relay *RPD FA C 911* deactivated) at the same time that the turnout is in the left position, allowing the train to change tracks (relay *KAG a G* deactivated). In this configuration, the relay *PG 911* is deactivated, blocking the component *KIT C 911* to activate. Once the pedal does not detect the presence of a train, the relay *PG 911* is activated and the procedure may continue.

In this example, the relays *RPD FA C 911*, *KAG a G* and *INT AC V2* are used in order to prevent dangerous situations in case of dysfunctional failure, i.e., when a component has a malfunction. The functional state of these components are the deactivated state, which means that in a case of failures or lack of energy, the relays *RPD FA C 911*, *KAG a G* and *INT AC V2* will indicate, respectively:

- That there is a train in the dangerous zone, even if there is not;
- That the turnout is switched so a train may change the track, even if it is not true;

- That a part of a train is still in the dangerous zone, even if it is not.

In a failure mode, the system may stop working for a time until an external intervention. Although it is not the most functional procedure, it is the most safe, since no train will be allowed to move if the system detects that it may cause a collision. This is one of the reasons relay-based RIS has been so successful in the last decades.

Although the case study used in this thesis finishes here, the system continues executing in order to lead the train in the opposite direction back to the upper track through the other turnout and then allow a train to enter in the dangerous zone in the normal direction. These operations are controlled by the other part of the relay diagram, which is responsible for controlling the Control Area C. This part of the system is responsible for identifying the temporary exceptional situation and control the turnout in this area so the train may change to the upper track. Furthermore, once the Control Area A informs the availability of the track or the end of the exceptional situation, the system in Control Area C may allow the entrance of a train in the normal direction. Although this part of the system is not presented as part of the case study, its analysis, formalisation and specification may be performed in a future work.

This case study is useful in this work since it allows the analysis of the possible occurrence of a frontal collision or a derailment.

Collision Avoidance

During the execution of the system existing in the Control Area A, a train may enter in the dangerous zone if the component *KIT C 911* is activated. It is clear that the activation of this component may imply the extension of the track towards the opposite direction so a train in the Control Area A may pass. In this situation, one must guarantee that this train will not collide with a train that may enter in dangerous zone in the normal direction.

The permission for a train to enter in the dangerous zone through the Control Area C is given by the RIS existing in this area. However, this permission can only be granted if Control Area A informs the availability of the track, i.e., the train in the opposite direction cannot enter in the dangerous zone. This message is sent by Control Area A through the component *EF11*. So, the track can only be used in the normal direction by the control Area C if, and only if, this is indirectly allowed by Control Area A.

As a way to avoid a frontal collision, one may guarantee that the components *KIT C 911* and *EF11* are never activated at the same time, so the tracks cannot be used in both directions at the same time.

Derailment Avoidance

As presented before, many situations may cause a derailment in a relay-based Railway Interlocking System. Apart from the physical causes (components physical problems, like bent tracks, for instance), this problem may also be caused by errors in the system design. A faulty system may not be able to safely control the signals and turnouts or even predict and avoid problems caused by components malfunctions. In this context, it is crucial to guarantee that the relay-based RIS have a safe behaviour and that it is able to deal with every known malfunctions that may lead to accidents.

In the ITCS example, for instance, during the ITCS procedure, a derailment may be caused by the passage of the train over an unlocked turnout. The blockage of the turnouts is a known procedure in order to avoid these components to move when the train is in transit over it, which could cause the train to run out from the tracks. Indeed it is a safety measure which guarantees that a train does not pass over an unlocked turnout. However, due to an incompleteness of the relay diagram of this case study, it is not possible to guarantee this safety measure.

Nevertheless, although not presented in the diagram, the turnout locking is guaranteed by the SNCF specialists. Many components cannot be presented in only one relay diagram due to the space limitation. Some of the interlocking procedures are divided into many different relay diagrams and this is the case of the turnout locking in the ITCS example. So, this is one example of a lack of information due to the relay diagrams syntactical and semantic limitation. The analysis of the presented solution for the ITCS problem will thus conclude that the behaviour that can be deduced from this relay diagram is faulty and needs revision in order to avoid a derailment.

In fact, it is clear that the system analysis based only on the information presented inside the relay diagrams is not complete. The real position of the train cannot be detected based on the relay diagram, so it is not possible to define when a train is indeed in transit over the turnout. As a consequence, the knowledge about the system background and environment is crucial in order to be able to give a safety guarantee.

Anyhow, the use of a more complete and modern methodology for the system specification may be beneficial to the railway companies. The manual analysis of the relay-based RIS is not a satisfactory approach for a critical system. Instead of using relay diagrams for the structural modelling of the electrical circuits, one may use formal specification languages in order to structurally and behaviourally specify these systems. Formal specification approaches like the B-method, for instance, is grounded in mathematical foundations and is supported by many tools that allow the specification, analysis, verification and the automatic refinement and code generation. All these features may be used as a way to completely prove the system safety and transform these relay-based systems into computer-based RIS by refinement.

1.3 Formal Methods and Mathematical Foundations

One way to improve the quality of a system is to change the way it is documented. Many existing methods of documentation are sometimes inefficient, imprecise or ambiguous. An alternative that solves this problem is the use of Formal Methods, which is grounded in elementary mathematics and is able to produce precise and unambiguous documentation [Woodcock and Davies 1996]. This section focuses on presenting some of the capabilities of the formal specification methodologies as well as some of the mathematical foundations used and the B-method, a formal specification language that has been successfully applied in the railway industry.

1.3.1 Formal Specification Methodologies

Formal Methods use mathematical definitions in order to help in the documentation, specification, design, analysis and certification of computer software and hardware [Rushby 1995]. The mathematical rigour of these methods allows the analysis and verification of the models at

any part of the development life-cycle: requirements engineering, specification, architecture, design, implementation, testing, maintenance and evolution [Woodcock, Larsen, et al. 2009]. Furthermore, Formal Methods are known by their analytical techniques relying on mathematical models that allow the exclusion of design errors in hardware [Black et al. 1996].

The mathematical basis for Formal Methods has the same purpose as the ones for others engineering areas: *"add precision, to aid understanding, and to reason about properties of a design"* [Woodcock and Davies 1996]. However, while Aeronautics and Nuclear Engineering are grounded, respectively, on Fluid Dynamics and Quantum Mechanics, for instance, the basis of Computer Science is Mathematical Logic. This basis provides precise interpretation for some notions like, consistency, satisfiability and implementation [Rushby 1995], which are important for the software development.

Formal methods may be applied at the development of any system and it may benefit many areas [Hall 1990]. Moreover, in critical situations, increasing the level of formality may be necessary and a completely formal proof may be even required in some situations [Woodcock and Davies 1996]. The specification of a system allows one to prove properties about it, like consistency, or completeness. Furthermore, it is possible to guarantee that the system meets determined requirements, like safety or security.

Nowadays, it is possible to find many different formal languages applied to different types of systems. CSP (communicating Sequential Processes) [Schneider 2000] is one example of a formal specification language focused on the specification of concurrent systems and the patterns of interaction between them. Petri Nets [Petri 1962] is a formal approach for the development of concurrent and distributed systems where the system is specified in a graphical notation. Besides, this language is based on a mathematical theory that allows its use for some formal proofs [Sun 2015]. Regarding the industrial use of formal specification methodologies, the B-method [J.-R. Abrial, Lee, et al. 1991] has been successful. One of the reasons of this success is because the B-method disposes of a complete development methodology that begins in the abstract system specification until its implementation based on the system refinement.

In this context, aiming at the development of safe Railway Interlocking Systems, the use of Formal Methods is strongly recommended [CENELEC 2011]. In fact, as they are critical systems, the formal specification mathematical foundations can be used in order to prove the system safety. One example of a formal language that has been successfully used in industry for the development and verification of railway systems is the B-method. It has been used in projects like METEOR [Behm et al. 1999], COPPILOT [Lecomte, Servat, Pouzancre, et al. 2007] and SACEM [Guiho and Hennebert 1990], for instance. Furthermore, it is known to be one of the strongest approaches for the development of railway systems [Fantechi, Fokkink, and Morzenti 2013]. Some of the mathematical basis of B-method are Propositional and First Order Logic and Set Theory, which are presented in the sequel.

1.3.2 Propositional and First Order Logic

"Logic is one of the oldest intellectual disciplines in human history" [Suppes 1999]. It can be used to state observations, define concepts and formalise theories. In computer science, Logic may be utilised in order to prove mathematical theorems, validate engineering designs and diagnose failures, for instance. Logic is divided into several branches focused in different logical aspects.

Propositional Logic, for example, is concerned with propositions and their relationships.

A proposition is one statement about the world that may be either true or false. The Propositional Logic can be used as a way to combine and relate these propositions so one may express specifications and reason about them [Schneider 2001]. A proposition may be simple or compound. The former contains only one statement, while the latter can be formed by more than one sentence. Some examples of propositions are:

- "The relay r is activated";
- "The system is safe";
- "The block b is electrified and activated".

Propositional sentences can be compound by using five different operators: negations, conjunctions, disjunctions, implications and biconditionals. A compound sentence is also true or false and it can be formed by the combination of a simple sentence and an operator, two sentences and an operator, or many sentences and operators. Every operator gives a different meaning to the sentence. A negation of a sentence P , for instance, is written as $\neg P$ ("not P ") and it inverts the Boolean value of P , i.e.:

P	$\neg P$
true	false
false	true

Thus, if P is the sentence "The relay r is activated", $\neg P$ must have the meaning of "The relay r is deactivated".

A conjunction between two sentences P and Q results in the compound sentence $P \wedge Q$ (" P and Q "). This compound sentence can only be true if P and Q are true:

P	Q	$P \wedge Q$
true	true	true
true	false	false
false	true	false
false	false	false

If P and Q mean "The relay r is activated" and "The relay s is activated", respectively, a component whose activation depends on the activation of r and s will be activated only when P and Q is true, i.e., $P \wedge Q$.

A disjunction between the sentences P and Q is written as $P \vee Q$ (" P or Q "). This sentence is true if either P is true or Q is true, being false otherwise:

P	Q	$P \vee Q$
true	true	true
true	false	true
false	true	true
false	false	false

Following the same example, if the safety of a system depends on the activation of at least one of the relays (r or s), one must consider that the system is safe when $P \vee Q$ is true.

It is also possible to make an implication between the sentences P and Q , which is written as $P \Rightarrow Q$ (" P implies Q "). This compound sentence can only be false if P is true and Q is false.

P	Q	$P \Rightarrow Q$
true	true	true
true	false	false
false	true	true
false	false	true

In the relays example, if s is always activated when r is activated, this can be expressed as $P \Rightarrow Q$, since it is falsified when r is activated (P is true) and s is deactivated (Q is false).

However, in a situation where $P \Rightarrow Q$ and $Q \Rightarrow P$ are both sentences that describe a determined situation, one may write $P \Leftrightarrow Q$ (" P if and only if Q "), which is called as a biconditional or equivalence. This sentence is true either if both P and Q are true or if they both are false:

P	Q	$P \Leftrightarrow Q$
true	true	true
true	false	false
false	true	false
false	false	true

Following the same example, if r is activated when s is activated and vice-versa (another component always activates them together), one may conclude that $P \Leftrightarrow Q$. If this is a condition for the system safety and there is the possibility of one of these relays to activate without the other, the system may not be considered safe.

In this work, simple and compound sentences are used in a way to describe properties about the system. As each operator has a unique and well established meaning, the properties about a system may be verified during its execution in order to guarantee that they are respected. The safety of the system may be validated based on a logical verification of well defined safety properties. *"Propositional Logic does a good job of allowing us to talk about relationships among individual propositions, and it gives us the machinery to derive logical conclusions based on these relationships"* [Suppes 1999].

However, Propositional Logic is inadequate to express more general conditions. If one may express that every functional relay is activated if, and only if, it is electrified, the propositional logic fails because it does not provide means to make general assumptions. In this context, First Order Logic extends the Propositional Logic by allowing the definition of variables and quantifiers.

By using First Order Logic one may define logical, relational or quantified sentences. A logical sentence is the one defined in propositional logic, which can be made using the negation, conjunction, disjunction, implication and biconditional operators. A relational sentence is defined in the form of:

$$q(a, b);$$

where q is a constant related to the variables a and b . A relational sentence is analogous to a proposition in the Propositional Logic, since it can either be true or false. It is defined based

on the relation between variables that represent objects from determined sets. For instance, we may assume that the relation q determines that a relay belongs to an electrical circuit. So, $q(a, b)$ is true every time any relay (represented here by the variable a) belongs to an electrical circuit (represented by the variable b). In fact, a relational sentence may have one or several variables and it gives a characteristic about them by relating them.

A quantified sentence makes use of quantifier operators in order to define properties for the defined variables. A universally quantifier (\forall) is used in order to define that all objects of each variable group has a certain property. In this case, an implication is generally used inside the universally quantified sentence, which has the form:

$$\forall a, b. ((o(a) \wedge p(b)) \Rightarrow q(a, b)). \quad (1.1)$$

In this context, $o(a)$ is a relational sentence defining that the variable a is a relay, while $p(b)$ is another relational sentence stating that the variable b is an electrical circuit. Thus, this complete quantified expression is true when, for every component a and b , if a is a relay and b is an electrical circuit, a belongs to b ($q(a, b)$).

In the same format, one may define an existentially quantified sentence. An existentially quantifier (\exists) defines that at least one object of each variable group has a certain property. In this case, a conjunction is generally used inside the sentence, which has the form:

$$\exists a, b. ((o(a) \wedge p(b)) \wedge q(a, b)). \quad (1.2)$$

This expression is true if there exists at least one a and b that belong to the relay and electrical circuit groups, respectively, where a is part of the electrical circuit b .

First Order Logic enriches the Propositional Logic concepts by allowing the definition of more general properties. Furthermore, it gives initial concepts of sets and relations by defining variables that represent elements from a group (set) and relations between these elements that define properties about them. As well as Logic, Set Theory is a mathematical foundation that is widely used as a basis for many formal specification languages.

1.3.3 Basics of Set Theory and Relations

Basically, a set is defined as a collection of objects called *elements*. A known example is the set containing all natural numbers \mathbb{N} . Two sets A and B are equal ($A = B$) if they contain the exact same elements. In order to describe that an object x is an element of a determined set A , the notation " $x \in A$ " is used. Contrarily, the notation " $x \notin A$ " indicates that the object x does not belong to the set A . In this work, sets and elements are represented by strings of letters, numbers and underscores. However, sets are represented with the first letter in upper case format, while the elements representation does not contain upper case letters. Besides, some known sets have their own representation, like the sets \mathbb{B} and \mathbb{N} , containing Boolean values (true and false) and natural numbers, respectively. In this context, some true propositions are:

- $1 \in \mathbb{N}$,
- $true \in \mathbb{B}$ or

- $-1 \notin \mathbb{N}$.

Sets may be explicitly defined by listing their elements using curly brackets, as in $A = \{1, 2, 3, 4\}$, which describes a set A containing the sequential numbers from 1 until 4. This same set may be written by set comprehension, which uses mathematical and logical notation in order to describe the set:

$$A = \{x | x \in \mathbb{N} \wedge x > 0 \wedge x \leq 4\}.$$

In this case, " $x \in \mathbb{N} \wedge x > 0 \wedge x \leq 4$ " is a formula in x that describes a property of x as a condition for the object to belong to the set A . In this context, the notation $x..y$ is a way to implicitly define a set containing the numbers between x and y :

$$x..y = \{z | z \in \mathbb{N} \wedge x \leq z \wedge z \leq y\}.$$

Besides sets of numbers and Boolean values, the Set Theory supports the description of sets containing any type of objects. For instance, one may define the sets of components, relays and contacts as, respectively:

$$\text{Components} = \{\text{relay1}, \text{relay2}, \text{contact1}, \text{contact2}\},$$

$$\text{Relays} = \{\text{relay1}, \text{relay2}\},$$

$$\text{Contacts} = \{\text{contact1}, \text{contact2}\}.$$

Sets may be related to each other and manipulated. The notation $A \subseteq B$ implies that all the elements of A are elements of B , i.e., $\forall x.(x \in A \Rightarrow x \in B)$. In the components example, it is possible to state that $\text{Relays} \subseteq \text{Components}$, for instance. Besides, the set of all possible subsets of the set B can be denoted as $\mathbb{P}(B)$, known as the *power set* of B . The notation $A \cup B$ represents the union of the sets A and B , resulting in a unique set containing the elements of both sets. Furthermore, the notation $A \cap B$ represents the intersection of these sets, whose result contains only the elements that belong to both sets. So, it is possible to state that $(\text{Relays} \cup \text{Contacts}) = \text{Components}$ and $(\text{Relays} \cap \text{Components}) = \text{Relays}$, for instance. The number of elements of a finite set is defined by its cardinality, which can be expressed by the notation $\text{card}(A)$, where A is a set.

All these notations are resumed in the Table 1.3. Based on these notations, given the sets Components, Relays and Contacts defined before, and the sets $A = \{1, 2, 3, 4\}$ and $B = \{4, 5, 6\}$, some true propositions are:

- $A \cup B = \{1, 2, 3, 4, 5, 6\}$,
- $\text{Components} \cup \text{Relays} = \text{Components}$,
- $\text{card}(A) = 4$,
- $\text{card}(\text{Components}) = 4$,
- $(A \cap B) = \{4\}$,
- $(A \cap B) \subseteq B$,

- $Relays \in \mathbb{P}(Components)$,
- $A \cap B \in \mathbb{P}(B)$.

Table 1.3 – Set notations.

$SetA \cup SetB,$ $SetA \cap SetB$	Union and Intersection between the sets $SetA$ and $SetB$, respectively
$\mathbb{P}(SetA)$	Power set of the set $SetA$
$el_a \in SetA$	Membership notation indicating that el_a is an element of the set $SetA$
$card(SetA)$	Cardinality of the set $SetA$

As anything may be an element, pairs of elements may also be elements. Thus, it is possible to define relations between them. In this work, relations are represented as sets with the indication " \leftrightarrow ". In the set of natural numbers, it is possible to define a relation between the numbers and their squares as $Square_{\leftrightarrow} = \{(x, y) | x \in \mathbb{N} \wedge y \in \mathbb{N} \wedge x = y^2\}$. The set containing all possible pairs between the elements of the sets A and B is denoted as $A \times B$. This set is also called as the Cartesian Product and it represents the set $\{(a, b) | a \in A \wedge b \in B\}$. For instance, the Cartesian product between the sets $Relays$ and $Contacts$ may be defined as:

$$Relays \times Contacts = \{(relay1, contact1), (relay1, contact2), (relay2, contact1), (relay2, contact2)\}.$$

Based on the Cartesian product, it is possible to define what is a relation between sets. The set containing all the possible relations between the sets A and B ($A \leftrightarrow B$) may be defined as all the possible subsets of $A \times B$:

$$A \leftrightarrow B = \mathbb{P}(A \times B).$$

In this context, it is possible, for instance, to establish that $Square_{\leftrightarrow} \in (\mathbb{N} \leftrightarrow \mathbb{N})$. In a relation $R_{\leftrightarrow} \in A \leftrightarrow B$, the elements of A that are related to elements of B are elements of the relation domain ($dom(R_{\leftrightarrow})$). Contrarily, the elements of B that are related to elements of A are elements of the relation range ($ran(R_{\leftrightarrow})$). These sets are mathematically described as:

$$\begin{aligned} dom(R_{\leftrightarrow}) &= \{a | a \in A \wedge \exists b. (b \in B \wedge (a, b) \in R_{\leftrightarrow})\}, \\ ran(R_{\leftrightarrow}) &= \{b | b \in B \wedge \exists a. (a \in A \wedge (a, b) \in R_{\leftrightarrow})\}. \end{aligned}$$

A function is a type of relation in which the elements of the domain relate with, at most, one element of the range. Functions may be differentiated into partial or total functions. In this work, functions are represented as sets with the indication " \mapsto " for partial functions and " \rightarrow " for total functions. According to the definition, in the function $F_{\mapsto} \in (A \mapsto B)$, each element of A may appear in only one pair of elements, which may be described as:

$$\begin{aligned} A \mapsto B &= \{x | x \in A \leftrightarrow B \wedge \forall a, b1, b2. ((a \in A \wedge b1 \in B \wedge b2 \in B) \Rightarrow \\ & ((a, b1) \in x \wedge (a, b2) \in x) \Rightarrow b1 = b2)\}. \end{aligned}$$

This function is called as *partial function* because some elements of A may not be related to an

element of B . In a total function $F_{\rightarrow} \in (A \rightarrow B)$, the domain must be the same set as A , which may be mathematically described as:

$$A \rightarrow B = \{x | x \in A \rightarrow B \wedge \text{dom}(x) = A\}.$$

In order to work with a part of a relation or function, it is possible to restrict its domain or range to a set of elements. The domain restriction of a relation R_{\leftrightarrow} to a set A ($A \triangleleft R_{\leftrightarrow}$) of elements results in a relation containing only the pairs whose first element is an element of A :

$$A \triangleleft R_{\leftrightarrow} = \{(a, b) | (a, b) \in R_{\leftrightarrow} \wedge a \in A\}.$$

Similarly, the range restriction of a relation R_{\leftrightarrow} to a set B ($R_{\leftrightarrow} \triangleright B$) of elements results in a relation containing only the pairs whose second element is an element of B :

$$R_{\leftrightarrow} \triangleright B = \{(a, b) | (a, b) \in R_{\leftrightarrow} \wedge b \in B\}.$$

In opposition to the restrictions, it is possible to make the domain and range anti-restrictions in order to restrict the domain or the range of a relation R_{\leftrightarrow} to all its elements except those inside a set A . The notation $A \triangleleft R_{\leftrightarrow}$ represents the domain anti-restriction of R_{\leftrightarrow} to A , resulting in a subset of the relation R_{\leftrightarrow} , without the pairs whose first element belongs to A . It can be logically defined as:

$$A \triangleleft R_{\leftrightarrow} = \{(a, b) | (a, b) \in R_{\leftrightarrow} \wedge a \notin A\}$$

Similarly, the notation $R_{\leftrightarrow} \triangleright B$ represents the range anti-restriction of R_{\leftrightarrow} to B , resulting in a subset of the relation R_{\leftrightarrow} , without the pairs whose second element belongs to B . It can be logically defined as:

$$R_{\leftrightarrow} \triangleright B = \{(a, b) | (a, b) \in R_{\leftrightarrow} \wedge b \notin B\}.$$

Given an element a that belong to the domain of a function F_{\rightarrow} , i.e., $a \in \text{dom}(F)$, it is possible to use the notation $F_{\rightarrow}(a)$ in order to obtain the element from the range set that is related to a . This notation is not allowed to be used when dealing with relations. Instead, one may use the relational image notation $R_{\leftrightarrow}[A]$ in order to obtain the set of elements from the range of the relation R_{\leftrightarrow} that are related to the elements contained inside the set A :

$$R_{\leftrightarrow}[A] = \{b | b \in \text{ran}(R_{\leftrightarrow}) \wedge \exists a. (a \in A \wedge (a, b) \in R_{\leftrightarrow})\}$$

The relational inverse of a relation R_{\leftrightarrow} (written as R_{\leftrightarrow}^{-1}) is the relation that contains the same pairs of elements as R_{\leftrightarrow} but with an inverted order, i.e., $\text{dom}(R_{\leftrightarrow}) = \text{ran}(R_{\leftrightarrow}^{-1})$ and $\text{ran}(R_{\leftrightarrow}) = \text{dom}(R_{\leftrightarrow}^{-1})$. The relational inverse of R_{\leftrightarrow} may be defined as:

$$R_{\leftrightarrow}^{-1} = \{(a, b) | (b, a) \in R_{\leftrightarrow}\}$$

Based on the notion of function, it is possible to define a sequence of elements from a set A as a function from the set of positive natural numbers to A . In this context, the set containing all

the non-empty sequences from the set A can be defined as:

$$SEQ1(A) = \left(\bigcup_{n=1}^{\infty} (1..n \rightarrow A) \right) - \emptyset,$$

which describes the union of all total functions between a segment of \mathbb{N} (from $n = 1$ until $n = \infty$) and the elements of the set A , excluding the empty set (\emptyset). Furthermore, a set containing all the non-empty injective (with no repeated element) sequences from the set A can be defined as:

$$ISEQ1(A) = \{x | x \in SEQ1(A) \wedge x^{-1} \in (A \rightarrow \mathbb{N})\}.$$

So, a sequence S_{\rightarrow} of elements from a set A is defined as $S_{\rightarrow} \in SEQ1(A)$ if the elements can be repeated and as $S_{\rightarrow} \in ISEQ1(A)$ if there is no element repetition inside the sequence.

In order to extract elements from a sequence S_{\rightarrow} , some notations may be used:

- $first(S_{\rightarrow})$ gives the first element that appears in the non-empty sequence S_{\rightarrow} ,
- $tail(S_{\rightarrow})$ gives the sequence S_{\rightarrow} without the first element,
- $last(S_{\rightarrow})$ gives the last element of the non-empty sequence S_{\rightarrow} and
- $front(S_{\rightarrow})$ gives the sequence S_{\rightarrow} without the last element.

A summary of the relation, function and sequence notations is presented in Table 1.4. Set theory may be applied in the Formal Methods domain for the specification of many different systems. As relay diagrams are represented in a graph format, Set Theory may be used in order to define and represent graphs, which can be used later as a way to prove safety properties about the relay-based RIS.

1.3.4 Graph Description Based on Set Theory

As it is presented in [Trudeau 1994], Graph is a representation of a set of points (as objects) and how they are joined up. A graph may depict electrical circuits or road maps for instance. As this structure may be represented graphically, it may support one to understand its properties [Bondy and Murty 1976]. Figure 1.12 depicts a graph representation of an electrical circuit, where the electrical components and wires are depicted as vertices and edges. respectively.

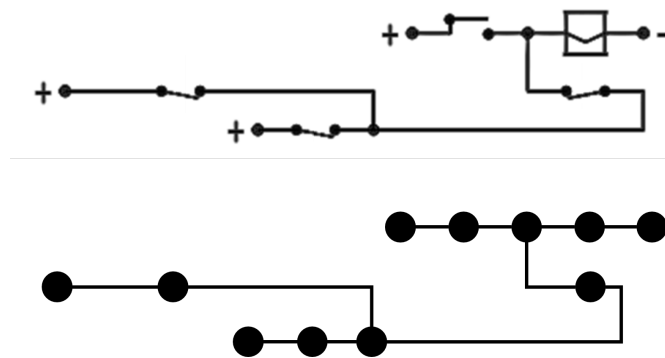


Figure 1.12 – An example of an electrical circuit and its graph representation.

Table 1.4 – Relation, function and sequence notations.

$SetA \times SetB$	Cartesian product between the sets $SetA$ and $SetB$
$Rel_{\leftrightarrow} \in (SetA \leftrightarrow SetB)$	Relation Rel_{\leftrightarrow} between the sets $SetA$ and $SetB$
$TF_{\rightarrow} \in (SetA \rightarrow SetB),$ $PF_{\leftrightarrow} \in (SetA \leftrightarrow SetB)$	Respectively, a total function TF and a partial function PF from the set $SetA$ to $SetB$
$Rel_{\leftrightarrow}[SetA]$	The relational image of the set $SetA$ in the relation Rel_{\leftrightarrow}
$SetA \triangleleft Rel_{\leftrightarrow},$ $SetA \triangleleft\!\!\triangleleft Rel_{\leftrightarrow},$ $Rel_{\leftrightarrow} \triangleright SetA,$ $Rel_{\leftrightarrow} \triangleright\!\!\triangleright SetA$	Respectively, the relation Rel_{\leftrightarrow} domain restriction, domain anti-restriction, range restriction and range anti-restriction to the set $SetA$
$Rel_{\leftrightarrow}^{-1}$	The relational inverse of the relation Rel_{\leftrightarrow}
$Seq_{\rightarrow} \in SEQ1(SetA)$	Non-empty sequence Seq_{\rightarrow} of elements of the set $SetA$
$Seq_{\rightarrow} \in ISEQ1(SetA)$	Non-empty injective sequence Seq_{\rightarrow} of elements of the set $SetA$
$first(Seq_{\rightarrow}), last(Seq_{\rightarrow}),$ $tail(Seq_{\rightarrow}), front(Seq_{\rightarrow})$	Given a non-empty sequence S , these notations denote: the first element, the last element, the sequence S without the first element and the sequence S without the last element, respectively.

A graph is considered directed when the edges are represented as arrows, indicating a direction from the vertex they begin to the vertex they end. Otherwise, the graph is considered undirected. Both graph types have their meanings and properties to be studied. As an example, while the graph in Figure 1.12 represent the structure of the electrical circuit, the graph in 1.13 represents the electricity flow in this circuit.

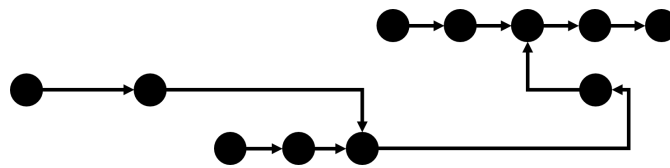


Figure 1.13 – Directed graph representing the electricity flow inside an electrical circuit.

According to [Bondy and Murty 1976], a graph is an ordered triple $(V(G), E(G), \psi_G)$, where:

- $V(G)$ represents a set of vertices,
- $E(G)$ represents a set of edges and
- ψ_G represents the incidence function that associates each edge to a pair of vertices.

In this work, based on Set Theory, it is possible to define the graph triple as $(Vert, Edg, Incid_{\rightarrow})$, being $Vert$ the set of vertices, Edg the set of edges and $Incid_{\rightarrow}$ the incidence function. In this

context, the incidence function can be defined as:

$$Incid_{\rightarrow} \in (Edg \rightarrow (Vert \times Vert)),$$

i.e., a total function from the set of edges to the set containing all the possible pairs of vertices.

In fact, as pairs of elements have an original ordered meaning, this graph definition based on Set Theory relations represents a directed graph. However, it is possible to work with this graph as an undirected graph by considering the vertices pairs and its relational inverse when defining the graph properties. This possibility may be observed during the definition of a graph path, for instance:

$$\begin{aligned} Path(va, vb, Incid_{\rightarrow}) = & \{Seq_{\rightarrow} | Seq_{\rightarrow} \in ISEQ1(Vert) \\ & \wedge first(Seq_{\rightarrow}) = va \wedge last(Seq_{\rightarrow}) = vb \wedge \\ & \forall i1, i2. ((i1 \in dom(Seq_{\rightarrow}) \wedge i2 \in dom(Seq_{\rightarrow}) \wedge \\ & i2 = i1 + 1) \Rightarrow ((Seq_{\rightarrow}(i1), Seq_{\rightarrow}(i2)) \in \\ & (ran(Incid_{\rightarrow}) \cup (ran(Incid_{\rightarrow}))^{-1})))\}. \end{aligned}$$

In this case, a path from a vertex variables va to vb is described as an injective non-empty sequence that begins in va and finishes in vb , where every couple of sequential elements is connected by an edge.

Similarly, a cycle may be defined as:

$$\begin{aligned} Cycle(va, Incid_{\rightarrow}) = & \{Seq_{\rightarrow} | Seq_{\rightarrow} \in SEQ1(Vert) \\ & \wedge first(Seq_{\rightarrow}) = va \wedge last(Seq_{\rightarrow}) = va \wedge \\ & card(Seq_{\rightarrow}) > 2 \wedge \\ & (\forall i1, i2. ((i1 \in dom(Seq_{\rightarrow}) \wedge i2 \in dom(Seq_{\rightarrow}) \wedge \\ & i2 = i1 + 1) \Rightarrow ((Seq_{\rightarrow}(i1), Seq_{\rightarrow}(i2)) \in \\ & (ran(Incid_{\rightarrow}) \cup (ran(Incid_{\rightarrow}))^{-1})))) \wedge \\ & (\forall j1, j2. ((j1 \in dom(Seq_{\rightarrow}) \\ & \wedge j2 \in dom(Seq_{\rightarrow}) \wedge j1 > 1 \wedge j2 > 1) \Rightarrow \\ & Seq_{\rightarrow}(j1) \neq Seq_{\rightarrow}(j2))) \wedge \\ & (card(Seq_{\rightarrow}) = 3 \Rightarrow \\ & (Seq_{\rightarrow}(1), Seq_{\rightarrow}(2)) \in ran(Incid_{\rightarrow}) \wedge \\ & (Seq_{\rightarrow}(2), Seq_{\rightarrow}(3)) \in ran(Incid_{\rightarrow})))\}. \end{aligned}$$

In this definition, a cycle in va is a non-empty sequence of elements that begins and finishes in va , where every couple of sequential elements is connected by an edge. Besides, the sequence must have more than two elements, every element must be different, except for the first and last, and if the sequence contains only 3 elements, it must be guaranteed that the two edges are different.

Based in this mathematical graph formalisation, one may also define other specific graph

notions, like the vertex degree, for instance. The degree of a vertex is the number of edges that are connected to this vertex, which may be mathematically described as:

$$\text{degree}(\text{vertex}, \text{Incid}_{\rightarrow}) = (\text{card}(\{\text{vertex}\} \triangleleft \text{ran}(\text{Incid}_{\rightarrow})) + \text{card}(\text{ran}(\text{Incid}_{\rightarrow}) \triangleright \{\text{vertex}\})),$$

where *vertex* is a variable that represents a vertex of the graph whose incidence function is *Incid*_→.

In order to provide a formal specification of relay-based Railway Interlocking Systems, this work proposes a formalisation of the information contained inside relay diagrams based on Propositional Logic, Set Theory and Graphs. Many formal languages support this mathematical background. Then, it is possible to perform an adaptation from the RIS formalisation to the system formal specification. One example of formal method that can be used in this context is the B-method, which has been successfully used in industry for the specification and proof of railway systems.

1.3.5 The B-method

The B-method [J.-R. Abrial, Lee, et al. 1991] is a formal method proposed by Jean-Raymond Abrial and published in details in 1996 in the book "*The B-book: Assigning Programs to Meanings*" [J.-R. Abrial 2005]. It defines the B-method as a "*model oriented*" method for specifying, designing, and coding software systems. It is close to VDM [Björner 1979a] and Z [Spivey and J. Abrial 1992] as some ideas of these methods can be recognised in B. This language focuses on the definition of abstract machines and their refinement until their implementation and code generation. In this context, refinement is the transformation from the abstract specification towards a more concrete one based on well-specified transformation rules aiming at the implementation of the system. more details can be found in [J.-R. Abrial 2005]. In B-method, a system implementation must provably be a refinement of a previously specified abstract machine.

The B-method abstract machines offer a structured approach that allows the management of large volume of detailed system description [Schneider 2001]. It is the basic building block of the language that allows a compositional approach where a large specification is constructed from smaller pieces. The combination of various abstracts machines is again an abstract machine. Furthermore, each abstract machine is divided in smaller parts, the clauses, each one describing different aspects of the specification.

A simple example of a B machine is presented in Figure 1.14. This example presents a machine that allows the storage of an information that can be either the answer yes or no. This small example is used throughout this section in order to demonstrate the usability of each clause. The notations used during the specification of an abstract machine are the basic mathematical notations for arithmetic, Logic and Set Theory, for instance. In order to be read by computers, these notations are generally transcribed to ASCII (American Standard Code for Information Interchange) character encoding, as presented in Table 1.5.

The first clause of a B-machine is the MACHINE clause, which defines the name of the machine. In a system specification, each machine must have a different name. The state of the system is defined in the form of variables inside the VARIABLES clause. Each variable must also have a different name. In the example of the Figure 1.14, the only defined variable is *answer*, which is


```

1- MACHINE
2   example
3
4- SETS ANSWERS = {yes, no, error}
5
6 VARIABLES answer
7 INVARIANT answer : ANSWERS & answer /= error
8 INITIALISATION answer :: {yes, no}
9 OPERATIONS
10
11   ans <-- set_answer(input) =
12   PRE input : ANSWERS & input /= answer & input /= error
13   THEN answer := input || ans := input
14   END
15
16 END

```

Figure 1.14 – Example of a simple B-machine.

responsible for storing the information.

Variables may be typed in terms of values, sets, relations, functions, sequences and other mathematical structures. These variables may be typed as natural numbers (NAT) or Boolean values (BOOL), for instance. Furthermore, new types may be defined inside the SETS clause in the form of sets of constant information. A variable typed as ANSWER, for instance, may assume the values yes, no or error. Conventionally, sets names are defined completely with letters in upper case format. The type of each variable must be assigned inside the INVARIANT clause. Moreover, any additional information about the variables must be defined inside this clause. The information specified as an invariant is treated as a set of properties that the system must meet during its execution. So this clause is used as a way to define any condition that must be met, like safety properties, for instance. As an example, the machine presented in Figure 1.14 contains an invariant specifying that the variable answer must never assume the value error, which could indicate a system malfunction. If the system execution may violate the conditions established in this clause, the machine is considered inconsistent, which may have different meanings depending on the type of information specified as invariant. If the invariant is a safety property and the system execution is inconsistent with it, the system may be considered unsafe, for instance.

The initial state of the machine must be provided inside the INITIALISATION clause. Every variable must be initialised, which must be consistent with the invariant. Then, the system state evolves with the execution of the system operations, which are specified inside the OPERATIONS clause. Each operation must have a specific name. The definition of inputs and outputs for the operation are optional. Every input is defined after the operation name inside the parenthesis and separated by commas. The outputs are placed before the operation name and followed by the notation <--. The operations must contain the preconditions after the reserved word PRE, specifying restrictions on parameters and limitations about when the operation may be called. In the machine example, for instance, a precondition for the set_answer operation is that the value of the input is different from the one stored by the machine and that the input value is not error. Once the operation precondition is satisfied, the operation body can be executed, changing the machine state by modifying the variables values.

The B-method accepts the use of non-determinism for the specification of abstract machines. Many non-deterministic operators are defined, being " : : " one example of them. This operator

Table 1.5 – B-method notations in unicode and ASCII [Schneider 2001].

Meaning	Unicode	ASCII
Disjunction	$P \vee Q$	P or Q
Conjunction	$P \wedge Q$	P & Q
Negation	$\neg P$	not(P)
Implication	$P \Rightarrow Q$	P => Q
Biconditional	$P \Leftrightarrow Q$	P <=> Q
Universally quantifier	$\forall x.(x \in T \Rightarrow P)$!x . (x : T => P)
Existentially quantifier	$\exists x.(P)$	#x . (P)
Equality	$E = F$	E = F
Set Union	$S \cup T$	S \/ T
Set Intersection	$S \cap T$	S /\ T
Member of	$e \in T$	e : T
Not member of	$e \notin T$	e /: T
Subset or equal	$S \subseteq T$	S <: T
Not subset or equal	$S \not\subseteq T$	S /<: T
Power set	$\mathbb{P}(S)$	POW(S)
Cartesian product	$S \times T$	S * T
Cardinality	$card(T)$	card(T)
Set comprehension	$\{x x \in S \wedge P\}$	{x x : S & P}
Empty set	\emptyset	{}
Assignment	$x := E$	x := E
Non-deterministic assignment	$x \in E$	x :: E
Parallel	$S T$	S T
Sequencing	$S ; T$	S ; T
Operation declaration	$v \leftarrow op(w)$	v <-- op(w)
Natural numbers set	\mathbb{N}	NAT
Boolean values set	\mathbb{B}	BOOL
Numbers from m to n	$m..n$	m..n
Domain of R	$dom(R_{\leftrightarrow})$	dom(R)
Range of R	$ran(R_{\leftrightarrow})$	ran(R)
Domain restriction	$U \triangleleft R_{\leftrightarrow}$	U < R
Domain anti-restriction	$U \triangleleft\!\!\triangleleft R_{\leftrightarrow}$	U << R
Range restriction	$R_{\leftrightarrow} \triangleright U$	R > U
Range anti-restriction	$R_{\leftrightarrow} \triangleright\!\!\triangleright U$	R >> U
Relational image	$R_{\leftrightarrow}[U]$	R[U]
Relational inverse	R_{\leftrightarrow}^{-1}	R [~]
Total function	$S \rightarrow T$	S --> T
Partial function	$S \mapsto T$	S +-> T

allows the assignment of any value inside a set to a variable. For instance, the initialisation `answer :: ANSWERS` arbitrarily assigns either yes or no to the variable `answer`. In abstract machines, there is not an operator for sequential actions, everything must be specified based on the parallel operator `"||"` between each pair of actions. However, in an implementation of an abstract machine, the parallel operator is not accepted and everything must be specified by means of the sequential operator `" ; "`. As non-determinism is also not accepted in an implementation, the refinement must replace the non-deterministic operators by the deterministic ones. A possible deterministic refinement of this initialisation may be, for instance, the explicit assignment of yes or no to the variable `answer`.

The B-method is supported by many tools that allow the specification, verification, analysis, refinement, implementation and code generation in order to produce systems that are correct by construction [McDonald and Anton 2001]. Some of the most used tools in industry are:

- Atelier B¹ [Clearsy 2011] - a set of tools that allow the development of systems using the B-method, offering an environment for the specification, refinement and code generation and making verification at each step based on the B-method proof obligations and its mathematical foundations;
- ProB² [Leuschel and Butler 2008] - a validation set of tools for the B-method, containing, for instance, a model checker, a refinement checker and an animator, which allows the animation of models as a way to allow the analysis of the system execution;
- Rodin³ [Butler and Hallerstede 2007] - a tool for modelling and verification of the B-method and an extension from it called Event-B [J.-R. Abrial and Hallerstede 2007].

Atelier B and ProB are some of the main tools that have been used in industrial projects as a way to support the use of the B-method and its mathematical foundations for the systems development and verification.

Regarding the industrial use of the B-method, some of the most important successful examples are the SACEM, METEOR and COPPILOT projects.

Delivered in 1989, the SACEM system was developed with the objective of being a computerised signalling system for controlling the RER line A in Paris [Bowen and Stavridou 1993]. The goal was to be able to increase the traffic movement by 15% maintaining the same safety levels of the conventional system. In this context, 63% of the system code was formally specified and verified, as this part of the system was considered as safety-critical [Guiho and Hennebert 1990]. In this case, *"the proofs were done interactively using automatically generated verification conditions for the code"* [Woodcock, Larsen, et al. 2009].

The SACEM system was the first industrial application of the B-method, which shows how this language was already mature at this time as it was able to contribute to the system safety.

After the experiences with the SACEM project, the capability of the B-method to be used for the development of large-scale industrial systems was proved with the development of the first driverless metro in Paris: METEOR. In Operation since 1998, the metro line number 14 was designed to manage the driverless trains in an environment mixed with manually driven trains. The safety-critical part of the system *"controls the running and stopping of every train, and controls opening and closing of doors located in trains and platforms"* [Behm et al. 1999]. Thus high safety levels was a strong requirement, together with high quality of service and easy operation management.

The use of the B-method in this project allowed this language to evolve and mature so it could be able to deal with such large-scale project. The METEOR development process is the natural development cycle of the B-method:

1. Modelling of the abstract models;

¹<https://www.atelierb.eu/>

²<https://www3.hhu.de/stups/prob/>

³<http://www.event-b.org/>

2. Refinement of the models until a concrete one that is able to be translated into code.

The concrete model was translated to Ada [Booch, Bryan, and Petersen 1994]. During the proof of the systems, errors were found and corrected. As a result, *"no bugs were detected after the proofs, neither at the functional validation, at the integration validation, at on-site test, nor since the metro lines operate (October 1998)"* [Lecomte, Servat, Pouzancre, et al. 2007].

The METEOR development demonstrated that the use of a formal development approach, like B-method, can be successful in a large industrial system development process. Furthermore, it made it clear that the complete proof of a complex system is feasible, where no bug is found after its development. Another important conclusion after this experience is that the use of Formal Methods can be cost-effective, since the METEOR development stayed within the initial budget and delay [Behm et al. 1999].

An experience of using the B-method in a different railway context was made in the COPPILOT project, which is an example of how formal specification methodologies may be used for the development and verification of other parts of the railway systems other than the signalling systems. In France, platform screen doors are used in order to avoid passengers to fall on tracks in many metro stations, which was adopted from the METEOR system. However, in order to be able to use these doors in stations where the trains are manually driven, a new system was required. In fact, as the trains could not be modified, the system should be able to detect their presence and open automatically the doors when the train is in the right position. An error in this system could lead to a risk for people lives.

The development of a SIL3 compliant system was made with the use of infrared and radar sensors and the B-method development cycle. The system was formally specified, proved, animated and refined until the generation of concrete models. The animation of the abstract specification allowed the designers to check the models against reality and verify their suitability. The concrete models were translated to the LADDER language. During the eight months system experiment controlling around 96.000 trains, no fault was observed.

Besides the successful use of this language in the industrial context, many academic researches have presented fruitful results on the use of the B-method in the railway field. Some important contributions in this field are the PERFECT and the NextRegio projects.

The PERFECT project [Ferlin et al. 2016] aimed at the creation of a global approach for implementing the ERTMS⁴ (The European Railway Traffic Management System) according to the country particularities. The proposed approach used high level Petri Nets [Peterson 1977] as a way to model the interlocking and signalling logic of the country, while the operating rules are modelled in UML. As Petri Nets and UML can both be translated to B, the global specification can be verified using the B-method supporting tools. A similar approach is provided by the NextRegio project [Ben Ayed, Collart-Dutilleul, and Prun 2016]. As a way to decrease the safety certification delay and efforts for single track low traffic French lines, this project proposes an approach for the system analysis focusing on the users needs as a way to produce a functional solution based on the requirements and the possible ways to fulfil them. As a way to verify the system, the interlocking and signalling systems are formalised in UML and then translated to B. Afterwards, the interlocking and operating rules are integrated, producing a complete model of the global system. The consistency of the model and the respect to the safety constraints are

⁴<http://www.ertms.net/>

verified by the Atelier-B tool.

Another example of academic work that uses the B-method in the development of railway systems is presented in [Sun, Collart-dutilleul, and Bon 2015], which proposes the transformation of the RIS models specified in Coloured Petri Nets [Jensen 1987] to B models. One of the objectives of this work is the use of the formal development process supported by the B-method refinement as a way to produce computer-based RIS. More details about this work is presented in the next chapter.

The B-method has been successfully applied at the development and verification of railway systems, but it has not been the only language used for this purpose. It is possible to find in the literature the usage of many different Formal Methods for the Railway Interlocking Systems safety proof and analysis, for instance. The next chapter presents an overview of these works, comparing their objectives with this present thesis.

Formal Specification of Railway Interlocking Systems

Contents

2.1 Introduction	48
2.2 Timetables-based Approaches	49
2.3 Relay and Computer-based RIS Formal Specification	51
2.3.1 Formalising Relay-based RIS Logic	52
2.3.2 Formal Specification and Implementation of Computer-based RIS	56
2.3.3 Other approaches	59
2.4 Conclusions	61

2.1 Introduction

The use of Formal Methods in the development of Railway Signalling Systems has been studied for many decades. Due to the critical aspect of these systems, many different efforts have been presented in the literature in order to use formal methodologies as a way to assure safety. Although the guarantee of the system safety is a common focus between all the approaches, the existence of many different formal languages capable of specifying different aspects of RIS results on the existence of multiple different approaches, each one dealing with different problems that must be addressed. In order to position this thesis in the literature, this chapter presents some of the existing works related to the use of Formal Method for the specification of Railway Interlocking Systems.

Regarding the application of formal techniques and tools to problems involving railway systems, [Bjørner 1998] is an important reference that lists 299 references separated by methods techniques and tools. This work is one evidence of the importance of using formal methods in the Railway domain in order to assure the system safety. As this list is from 1998, several new approaches and technologies have been studied and applied in the railway field. In this chapter many old and recent works that are close to the research context of this present thesis are discussed and differentiated.

Regarding the Railway Signalisation field, its safety-critical aspect is evident as the signalling systems control the trains movements, which is a task that has the potential to cause collisions and derailments if not performed with care. In order to guarantee the safety of these systems, many works have presented approaches for formally specifying them. Furthermore, with the technological advances, some approaches also proposed the use of formal development processes in order to implement these systems with computer-based technologies. In the RIS context, it is important to mention that there are two different levels of systems used for the interlocking safety: the Dispatcher and the Local Interlocking levels. Each one has a different view of the system and deals with different problems.

In the Dispatchers Level, the system has the objective of managing the trains timetables. So, this system is responsible for creating the trains routes as a way that they will never lead trains to collide with each other. More specifically, it guarantees that trains that use the same tracks or turnouts maintain a certain safe distance between each other. Once the collision absence is guaranteed, the system deals with the control of turnouts and level crossings when the trains are close to them. Furthermore, it must deal with emergence situations, being able to re-route trains when necessary. Thus, the dispatchers system is responsible for giving the train routes and keep track of them and their needs when they are following these routes. The systems in this level are often called as "functional", since a centralised computer-based system that stores the interlocking logic is responsible for controlling the whole interlocking operation and the trains routing. The physical geographical position of the components is not relevant for this type of system [Banci, Fantechi, and Gnesi 2004].

In the Local Interlocking Level, the system has the objective of controlling the electrical components in a way to ensure local safety. The interlocking systems are generally implemented by electrical circuits or a joint implementation of computer and electrical components. In this context, the RIS is responsible for detecting the presence of the train and controlling every track-side electrical component (turnouts, signals, antennas) in order to guarantee safety. These

systems are divided into small areas that have their own particularities that must be taken into consideration. Hence, they are often named as geographical, since they are responsible for small portions of the system, being able to be extremely specific to the track configuration and the geographical position of the electrical components [Banci, Fantechi, and Gnesi 2004].

In fact, the existence of these two systems is extremely important as a way to guarantee the RIS safety. The redundant overlapping safety measures is an approach that is indeed stimulated as a way to decrease risks. However, as the logic and the level of abstraction of these systems are extremely different, they are generally specified and developed separately. While one is centralised and focused on the train routing through a fixed rail network, the other is divided into small parts that deal with local specific problems and configurations. Thus, this chapter divides the existing approaches in the literature into two different subgroups according to the different interlocking system levels.

2.2 Timetables-based Approaches

Many formal methodologies for the specification of Railway Interlocking Systems are focused on the functioning of the systems present in the Dispatchers Signalling Level. So, they specify the logic of a system that is focused on the control of the trains routes, which generally is based on the timetables, control tables, routing tables or interlocking tables. Although this is not the context of this thesis, some of the works discussed in this section present some interesting methodologies that can be used as inspirations for proposing a solution to our problematic. Thus, it is important to position this thesis in the literature regarding the characteristics of these other works. Although the use of formal specification methodologies in the RIS context is the focus of this section, some more informal methodologies are also discussed due to their important and inspiring contributions. This is the case of [Xiangxian, Yulin, and hai 2011] and [She et al. 2007].

In order to specify an interlocking system, some approaches propose the separation between the specification of the interlocking rules from the system structure. The main reason for this separation is that the interlocking rules always remain the same, thus the system interlocking logic is determined by the physical structure, which must be analysed. In [Xiangxian, Yulin, and hai 2011], for instance, it is proposed the use of a component-based model for the description of the station topology, which can be associated automatically with the fixed interlocking rules by using software tools. Despite the difference of the RIS level, the separation between the system structural design and the interlocking rules can also be applied to the relay-based systems. Indeed, the structure of these systems is the only documentation presented in the form of relay diagrams. In order to formally specify these systems, the separation between structure and behaviour may be an interesting solution.

The specification of the system structure is also an interesting topic for discussion, as many works propose the use of traditional data structures for describing the physical system design. In order to represent the railway topology, a graph is an abstraction commonly used, as the railway network can easily be represented with this data structure. This is the case of [Xiangxian, Yulin, and hai 2011], which use non-oriented graphs in order to represent the connections between track sections, semaphores and turnouts. In this case, a graph is used in order to implement a tool for finding the trains possible routes and their compatibility. Similarly, the work presented

in [She et al. 2007] proposes the use of graphs for the railway topology description. By using a directed weighted graph it is possible to define an algorithm in order to find the shortest route between two places as well as define all the possible other routes in case the shortest one is not available during the system execution. The representation of the system structure as a graph allows the use of the native graph operations in order to reason about the system. Although a formal specification methodology is not the focus of these works, they illustrate how graphs and their supporting operations may be useful in order to abstract these systems.

Besides the graph structural representation of the system, many of the works in the dispatchers level use the control tables in order to represent the interlocking systems behaviour based on another structural representation of the system. These tables present all possible and required routes in a station derived from the signalling layout, so it is possible to apply formal methodologies in order to verify the control tables and identify conflicting routes. This is the case, for instance, of the work presented in [Mirabadi and Yazdi 2009], which proposes a tool for the generation and formal verification of the system control tables based on the signalling layout planner. In this context, the structural representation of the system supports the definition of the possible routes based on the knowledge about the system general behaviour. Similarly, [Winter et al. 2006] presents an approach where control tables are generated from the track layouts. Then, these control tables are used in order to produce a model of the system behaviour based on the positions of the trains on their routes.

The specification language used in order to model these systems is also an important point of discussion. Each language has different capabilities and focus in a way that the language used must be chosen wisely. CSP [Hoare 1978] [Schneider 2000], for instance, is a language for the specification of concurrent systems used for the specification of RIS in [Winter 2002]. This work proposes a formal model of the functional specification for a track layout and uses a signalling principles formal model in order to verify the system safety. The specification of the track layout is based on the control tables. Therefore, this work concludes that the process algebra that basis CSP is not very well suited for describing the content of control-tables as it does not yield a good documentation. Then, in [Winter et al. 2006], the authors propose the formal specification of the interlocking using Computational Tree Logic (CTL) [Emerson 1990]. The objective of this work is the improvement of a pre-existing toolset for the generation of control tables.

In the general signalling context, Petri Nets [Peterson 1977] is a well known language that is commonly applied in the system specification. This language allows modelling the system in a graphical form which is generally more comprehensible for non experts in Formal Methods. In [Khan, N. A. Zafar, et al. 2014], for instance, it is proposed to limit the responsibility of controlling the trains movements only to the trains. In this work, Petri Nets is used as a tool for the specification of the concurrency issues in the communication between the train and the track components (turnouts or level crossing agents). The routing map is then used in order to inform the upcoming components in the train route. Coloured Petri Nets (CPN) [Jensen 1987], an extension of this language, is also widely used, like in [Vanit-Anunchai 2010], which proposes the formal modelling and verification of control tables. In this latter example, one of the reasons of using Petri Nets is its high level of user friendliness, which may be helpful for control engineers to understand the system models.

As each language has the potential to allow the specification and verification of a different

aspect of the system, they may provide different and useful views of the system execution and safety. In the context of relay-based systems, the use of these languages may be studied in order to identify the possible benefits that each formalism may provide.

Regarding the verification of these functional systems, there is an important discussion about the use of model checkers. The application of model checkers in this context is a natural choice as the system safety is verified in order to avoid dangerous states. Nonetheless, this technology has a limitation regarding the size of the state space. In the dispatchers level, the verification of the system with a model checker tends to be costly as the specification contemplates the whole railway network, which is generally large. So, when specifying the railway topology, it is necessary to consider this issue. The work presented in [Winter 2002], proposes some solutions in order to reduce the number of states that the model checker has to investigate. Although it concludes that CSP is not well suited for the context of this work, it presents that the counter-examples provided by the FDR [Gibson-Robinson et al. 2014] model checker were useful and easy to understand, even for non-experts on CSP.

Model checking techniques are also used in [Mirabadi and Yazdi 2009] and [Ghosh et al. 2016] for the verification of control tables using the NuSMV model checker, but no discussion about the limitation of this tool is provided. Nevertheless, [Ferrari et al. 2011] reaffirms that the use of model checkers in this context may be difficult due to the high number of variables, which may cause a state space explosion. In this work, it is presented an analysis of the use of model checkers for the verification of Control Tables and one of its conclusions is that small interlocking systems can be addressed by model checking, but medium or large size interlocking systems cannot. [Ferrari et al. 2011] compares and analyses the performance of the NuSMV and SPIN [G. Holzmann 1997] model checkers in this RIS context. NuSMV is also used for the system formal verification in the work presented in [Winter et al. 2006], but it proposes a number of optimisations in order to minimise the state explosion problem and improve performance without losing credibility regarding safety issues. Thus, in order to verify these systems using model checkers, it is required to use alternative solutions in order to reduce the size of the state space. Nonetheless, in the context of this present thesis, the use of model checkers in the geographical context may still be analysed as these systems are generally concerned with local installations instead of a considerable railway track network.

A last work that is worth mentioning is the one presented in [Ghosh et al. 2016], which proposes a mixed approach between the functional and geographical view points of the system. This work presents a tool flow for the generation and verification of RIS safety properties as well as the prioritisation of acceptance tests. Control tables are used as a way to detect conflicting routes, but the safety properties are defined based on the states of the relays, signals and turnouts in a route. Despite its proximity with our relay-based context, this work is more focused on the generation and verification of control tables for a yard.

2.3 Relay and Computer-based RIS Formal Specification

The experiences of the use of Formal Methods for the specification of systems in the Dispatchers Signalling Level can inspire our work. However, as the focus of this present thesis is the formalisation of the systems in a geographical point of view, the analysis of the literature

regarding the works that propose the formal specification of local interlocking systems is extremely important in order to position this thesis. Nonetheless, the literature does not present many efforts on formalising relay-based systems, as they are generally being directly replaced by computer-based ones. As we also objective the creation of computer-based systems through the formal specification of the ancient relay-based ones, this section also presents many of the efforts on the use of formal methodologies for the creation of computer-based RIS as documented in the literature. Than, some extra approaches with unusual techniques and presenting contextual similarities with this present thesis are also discussed.

2.3.1 Formalising Relay-based RIS Logic

This section presents the works which have the most correlations with the context of this present thesis: the formal specification and verification of relay-based Railway Interlocking Systems. There are not many contributions in this field, since these ancient systems are being replaced by modern computer-based ones. So, the formalisation of the relay-based RIS are generally made with the objective of (1) proving the safety of the existing systems that still have not evolved to the new technologies or (2) use formal methods as a tool for the generation of safety-proved computer-based RIS by refinement based on the successful existing historical relay-based ones. This section discusses some of the most important works in this field, focusing on their contexts, objectives and propositions. Table 2.1 summarises the methodologies presented in this section, containing their industrial context, the formal method used and the verification methodology applied.

Table 2.1 – Methodologies presented in literature for the relay-based RIS formal specification.

Work	Industrial context	Formal method used	Verification methodology
[Haxthausen, Kjær, and Le Bliguet 2011] [Haxthausen 2013]	Danish systems	SAL and LTL	Model-checking
[Cavada et al. 2018]	Italian systems	SMDKN	Model-checking
[Sun, Collart-dutilleul, and Bon 2015][Sun 2015]	French systems	Petri-nets ^a	Model-checking
[Van Eijk 1997]	Independent models	PROMELA	Model-checking
[James, Lawrence, et al. 2013]	Ladder models	Logic	Model-checking

^aThis work also used B-method for the RIS specification, but not for the models related to the relay diagrams as discussed in the next section.

Regarding relay-based systems, the railway context is an important factor for comparing the works in this field, as each company in each country has its own design rules for modelling relay diagrams. The methodologies created for the formal specification of relay-based RIS are generally based on the relay diagrams that model the structure of these systems. However, as there is not a unified methodology for designing these diagrams, the approaches presented here have a tendency to be focused on their own contexts, being generally incompatible with the others. The Figure 2.1 present some relay diagrams used in some of these approaches.

In the Danish context, an important contribution to this field is made in [Haxthausen, Kjær,

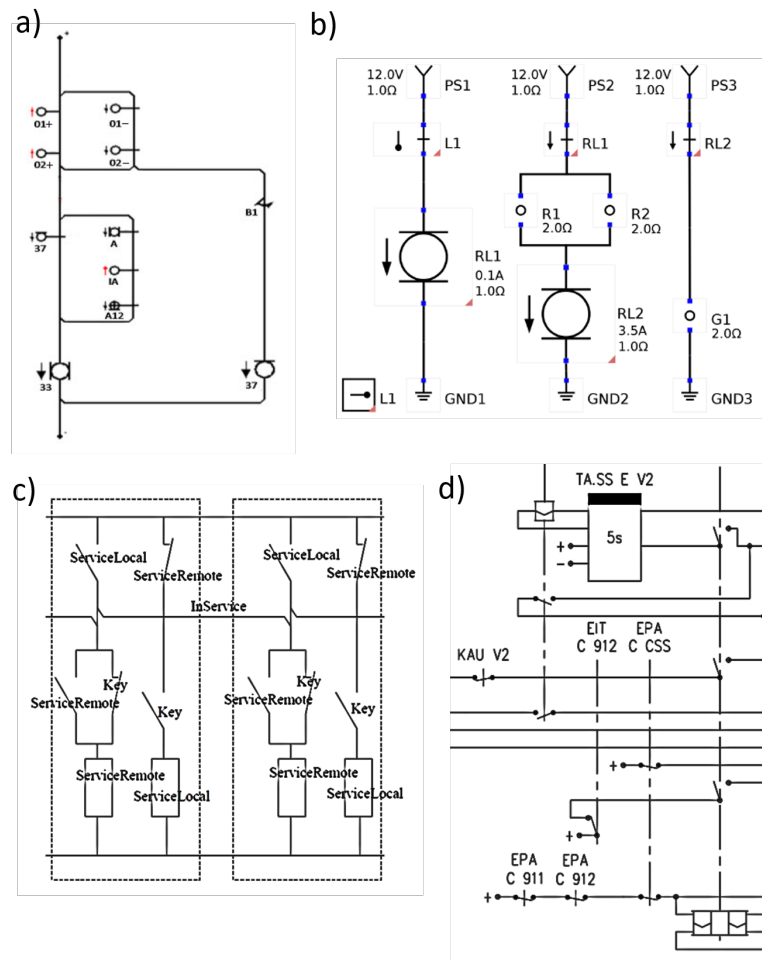


Figure 2.1 – Some relay diagrams modelling styles: (a) Danish [Haxthausen, Kjær, and Le Bliguet 2011], (b) Italian [Cavada et al. 2018], (c) Ladder-like [Van Eijk 1997], (d) French [Sun 2015].

and Le Bliguet 2011] [Haxthausen 2013]. These papers propose a set of tools for: the automatic generation of formal models based on the circuit diagrams of the RIS, and their automatic verification regarding safety requirements. The relay diagrams presented in these works for the Danish systems are quite simple: they contain energy sources, relays, contacts and buttons connected by wires. Moreover, the components states are indicated by arrows and the relays and contacts are related by having the same name.

A solution for the formal specification of the Italian relay-based RIS is proposed in [Cavada et al. 2018]. This work presents a methodology and a tool chain for the analysis of these systems. The relay diagrams in this Italian context present many details about the electrical current and resistors values. Furthermore, they contain many different components like energy sources, levers, relays, contacts and lights, for instance. As this work proposes the formalisation of a wider set of components, it presents a more complete approach that can be applied to other different contexts. As well as in the Danish systems, the relays and contacts are related by their names and the indication of the electrical components states is depicted by arrows.

An ancient discussion about the use of formal modelling and verification tools to support

the description of electromechanical relays circuits is presented in [Van Eijk 1997]. Differently of the other contributions presented in this section, the relay diagrams used as basis in this work are not designed by railway companies, in a way that they are graphically closer to ladder diagrams. Due to this lack of design rules, there is not much discussion about the limitations of this approach. Nevertheless, it presents an interesting initial debate about the possibility of using Formal Methods for the specification of relay-based RIS.

Ladder is still used by many companies in order to model the interlocking systems electrical circuits. A recent example of a work that proposes the formal specification of the logic used in these models is detailed in [James, Lawrence, et al. 2013]. This work presents how the systems modelled in ladder can be transformed into a formal specification based in propositional logic so it can be automatically verified according to safety properties. It is a successful example of applying formal specification into the development of industrial interlocking systems.

Working in the same context as this present thesis, a formal specification of the French Railway Interlocking Systems is proposed in [Sun, Collart-dutilleul, and Bon 2015][Sun 2015]. Although the focus of these contributions is the modelling of the system according to the position of the components in the tracks, they also proposes an event-based specification of the low-level systems (relay-based) in order to clarify their structure and to make them easier to be constructed. The french relay diagrams are complex as they support many different types of components, differentiating monostable and bistable relays and allowing the modelling of timed components. These characteristics make the French diagrams the most rich in terms of syntax and semantics, thus, methodologies used to formalise systems from other contexts can be difficult to be applied in the French one.

All these works are focused on their specific railway contexts. The execution of these systems are similar independently of the context, since they are focused on the electrification and activation of components. However, the variety of different components that can be modelled as well as the interpretation of the relay diagrams are some factors that must be considered when applying one of these methodologies in a different context. The creation of an approach offering the possibility to be adapted to different contexts can be beneficial as a way to establish a general common strategy, enabling its use by many different companies in order to formalise and verify the relay-based RIS.

The language used in order to model these systems is an important factor to be considered in the analysis of these approaches as it can impact on the properties that may be verified. Each language has a different focus and different characteristics that make them advantageous in determined aspects. Petri Nets, for instance, is a graphical language known for being clear and comprehensible so it can be understood by railway engineers. In the French context, Coloured Petri Nets is recognised by the French National Railway Company as a formal tool for scientific research [Lalouette et al. 2010][Buchheit et al. 2011]. The work presented in [Sun, Collart-dutilleul, and Bon 2015][Sun 2015] proposes the use of Coloured Petri Nets in order to model the high level functions of a RIS. Nonetheless, it also presents an approach for the specification of relay-based systems, supporting the specification of the system synchronisation based on the concurrency notations of the language.

The work presented in [Haxthausen, Kjær, and Le Bliguet 2011][Haxthausen 2013][Haxthausen 2014] make use of two different formalisms: the SAL language [De Moura, Owre, and

Shankar 2003] for modelling the RIS, and LTL (Linear Temporal logic) for the definition of the safety conditions. The former supports the specification of the system based on state transitions as a way to describe the relays states, which is the particular interest of this work. Then, LTL can be used to express all the required system properties over the states of these components.

A different approach for modelling the systems is presented in [Cavada et al. 2018], where the systems are reduced to a Switched Multi-Domain Kirchhoff Networks (SMDKN) [Janschek 2011]. This formalism allows the modelling of a network of components connected according to the Kirchhoff conservation laws. So, it enables the definition of the system state transition over time. Then, for each system configuration, the SMDKN behaviour is defined with a Differential Algebraic Equation derived from the components behaviour and the Kirchhoff laws. This formalism allows the analysis of the circuits at the physical level, supporting a comprehensive understanding of the railway control actions. This specification is more focused on the description of the system based on the required voltage and current for the components states.

In [Van Eijk 1997] the use of PROMELA [G. J. Holzmann and Lieberman 1991] is proposed as a formalism for the formal specification of the electrical circuits. This language was originally designed for modelling communicating finite state machines, but it can also be used for the specification of other parallel or distributed systems. Similarly to CSP, this language allows the system specification in the form of processes and the state succession is represented by the succession of events represented through channels. This allows one to model the electrical components state succession and analyse it.

The work presented in [James, Lawrence, et al. 2013] presents how the interlocking systems can be modelled in propositional logic. Thus this approach is close to our proposition of using pure logic as basis for modelling these systems. Although propositional logic is simpler than First Order Logic (considering that the latter is an extension of the former), it is still enough for modelling the ladder diagrams, which is only composed by energy sources, relays and contacts. Furthermore, as this work presents a precise formalisation of the transformation from ladder logic to logical mathematical expressions, this transformation can be automated, which is an interesting contribution that is worth to be mentioned.

Regarding the system formal verification, model checking techniques is used in [Sun, Collart-dutilleul, and Bon 2015][Sun 2015], but the space state is compacted by a simplification methodology, which allows a lighter verification without losing reliability. Model checking is also applied in [Haxthausen 2013] in order to verify that the defined safety conditions are satisfied by the model that describes the system states. In this work, the SAL model checker¹ [Moura, Owre, and Shankar 2003] is used and it does not report any problem regarding scalability issues. An SMT-based model checking is proposed in [Cavada et al. 2018]. In this work, it is demonstrated the practical scalability when using the model checking techniques in experimental evaluations. In [James, Lawrence, et al. 2013], it is also presented a successful use of model checking in order to verify these systems when verifying an industrial case study with different tools.

There is a wide tendency on using model checking for the verification of such relay-based RIS. The definition of the system behaviour based on the electrical components states is logical and model checkers are the perfect tool for the verification of these systems. Moreover the

¹<http://sal.csl.sri.com/>

works presented in this section provide examples of successful use of model checkers for the verification of such systems. Nonetheless, there is a lack of analysis in this field regarding the limitation of this technique in terms of state space size. In this context, it is still necessary to perform an evaluation of the use of this technique on the verification of different systems with different sizes. Furthermore, the use of other approaches of verification can still be analysed. For instance, one could use the B-method logical proof obligations in order to analyse the system consistency regarding the defined invariant. As there are not many contributions in this field, there are still not many works related to the investigation of the use of different verification methodologies in this context.

Some works that propose the formal specification and verification of RIS also present a methodology for the refinement of these models in order to produce computer-based RIS. This is the case of [Sun, Collart-dutilleul, and Bon 2015][Sun 2015], for instance. In this context, many documented approaches have the objective of formally specifying these systems from scratch based solely on their execution logic, using a formal development process for the generation of computer-based systems. The formal specification for the computer-based RIS implementation is discussed in the next section.

2.3.2 Formal Specification and Implementation of Computer-based RIS

The formal methodologies of specification may be used as a tool for the generation of safety-proved computer-based RIS. Some of the approaches already presented in the previous sections also objective the creation of computer-based systems, like [Ghosh et al. 2016] and [Sun, Collart-dutilleul, and Bon 2015], for instance. This section focuses on discussing the methodologies for the specification and implementation of computer-based Railway Interlocking Systems that control the interlocking components with the objective of avoiding unsafe situations. It is important to mention that, although some works use control tables or other models based on the railway network modelling as a way to support the system specification, they still support the verification of the local components behaviour (related to signals or turnouts). Thus, they are also detailed in this section.

The most important difference between the works presented in this section is the information about the systems that they use as basis for their formal specification. While in the context of this present thesis the relay diagrams are the most structured models that grounds the understanding of the system behaviour, in some other contexts the system are modelled by railway networks representations. This is the case of [Xiangxian, Yulin, and hai 2011], for instance, which proposes the use of graphs as a tool for abstracting the railway network and then generating the interlocking logic for the control of the interlocking components. The authors affirm that this is a more efficient solution to generate the interlocking logic, since this approach may be easily automated. However, although the use of graphs to support the system implementation is an interesting solution, [Xiangxian, Yulin, and hai 2011] does not present a strong mathematical background that could allow the safety verification of these systems.

A graph representation of the system structure is also used in [Hansen 1994][Hansen 1998a], which presents a first attempt in order to formally specify the Danish RIS. This work proposes an approach for the formal specification of RIS based on a graph representation of the tracks map. It considers the positions of elements like turnouts and signals in the tracks as a way to

define collision-avoidance safety requirements in VDM [Bjørner 1979b] and then validate the system through simulation. The use of a graph approach is an interesting solution in order to analyse the system as it allows the modelling of the railway network topology. Later, in [Hansen 1998b], this approach is improved so the existence of derailments can also be analysed.

The information contained inside the train routing maps are used in [Khan, N. A. Zafar, et al. 2014] in order to inform the upcoming components in a train route. As mentioned before, this work aims the development of a system where the train is the only responsible for controlling the track-side components so there is no conflict between two systems decisions. This is a completely different approach that uses Petri Nets and their concurrency aspects in order to specify the communication between the dispatchers office, the trains and the track electrical components. This language is also used in [Sun 2015][Sun, Collart-dutilleul, and Bon 2015], which proposes the use of its extension, Coloured Petri Nets, in order to formalise the French RIS. The focus of this approach is on the simulation of the RIS behaviour based on a specification of the system installations, signalling operations and rolling stock movements. Furthermore, it also proposes the formal specification of relay-based RIS from the existing relay diagrams. As a way to allow the implementation of these systems, this work proposes the translation of the CPN specifications to B, a language that supports a complete development process through refinement. However, it does not include the implementation of the existing relay-based systems as computer-based ones. The main objective of this work is to produce mixed systems which are referred as relay-based computer-controlled RIS.

The layout of the interlocking system is also used as basis for the geographical RIS formal specification in [Banci, Fantechi, and Gnesi 2004]. This work focuses on modelling each physical entity (turnouts and signals, for instance) based on their position in the yard topology. Then the union of the entities specification supports the description of the interlocking rules. The safety rules are defined in order to allow only certain combinations of components states as a way to avoid collisions. The RIS formal specification is made in statecharts [Harel 1987], which allows the description of the transitions between states.

The work presented in [Roanes-Lozano et al. 2011] proposes a simple use of logic in order to specify and prove the safety of local RIS. In this approach, a general railway topology is analysed based on a conceptual knowledge regarding the position of the trains and the safety conditions, ignoring the technical physical part of these systems. As the relation between the Boolean values and the physical electrical components are extremely simple in this work, it can operate in a higher level of abstraction than what is presented in other works. A specification based on the occupation of the trains on the tracks is also proposed in [Antoni 2009]. It objectives the reduction of testing time and cost during the production of SNCF computer-controlled RIS. Although this specification is based on a higher level of abstraction of the RIS, it aims the conservation of the relay-based RIS safety level when producing computer-based ones, which is an objective of this present thesis.

Interlocking tables are used as basis for the RIS formal specification in [Hei, Takahashi, Nakamura, et al. 2006][Hei, Takahashi, and Hideo 2008][Hei, Takahashi, and Nakamura 2009]. This work proposes the creation of Component-Based Decentralised RIS (CBDRIS), using component-based software development technology in order to allow the standardisation of hardware and software in such systems. As the interlocking tables describe the conditions for signals to change,

they are used in this work to support the definition of the standardised components. The safety and reliability analysis of the components in this work is performed using two types of extended Petri Nets: G-nets [Figueiredo and Perkusich n.d.] and Deterministic and Stochastic Petri Nets (DSPNs) [Zimmermann and Hommel n.d.][Marsan and Chiola 1987]. Petri Nets are also used in [Amendola et al. 1996] in order to represent the design measures adopted to detect faults and errors and prevent the error propagation within the Italian RIS. This work summarises an approach for the design, implementation and validation of computer-based RIS regarding the European safety integrity levels. Moreover it also proposes the generation of the system preliminary evaluations regarding their safety integrity. In this context, the system safety integrity is estimated based on conservative assumptions on the model parameters values,

The formal specification of the logic described in the system interlocking tables is also proposed in [Eris and Mutlu 2010]. This work analyses the use of three different formal methods for the development of a software algorithm for a signalling system: an Asynchronous Sequential Circuit, an Automaton-based and a Petri Net-based design model. It concludes that Automaton-based and Petri Nets models are the best solutions for the design of the railway signalling systems. This is because the asynchronous sequential circuit design method presents problems in the design and application phase of the system development. Although the main objective of this work is the evolution of the Turkish relay-based systems to Programmable Logic Controllers, it does not work in a direct translation from the logic behind the existing systems.

An unusual approach is presented in [Dipoppa et al. 2001], which proposes the integration of a formal verification tool in a preexisting design flow in order to ensure safety through automatic verification. For this purpose, this work proposes the use of two SAT-based model checkers, BMC [Biere et al. 1999] and SATO [Zhang 1997], in order to verify safety requirements given by an expert. As a way to make this integration cost effective, the verification is applied directly to the RIS program.

The methodologies for the refinement of the system specification are also an interesting discussion point as they can represent a deciding factor in order to apply these methodologies in industry. This is because some refinement approaches can be supported by tools that ease the process. Furthermore, some specification languages have a small number or even no step for the implementation of these systems. This is the case of the work presented in [Hansen 1994][Hansen 1998a] [Hansen 1998b], whose specification language can be executed using the VDM tool-box [Elmstrøm, Larsen, and Lassen 1994]. As a consequence, the refinement of the models towards the creation of an implementation was not necessary. Although the use of specification as implementation is discouraged [Hayes and Jones 1989], this work mentions the reduction of the number of models and refinement proofs as an important advantage of this approach.

The work presented in [Sun, Collart-dutilleul, and Bon 2015][Sun 2015] proposes the transformation of the RIS formal specification made in Coloured Petri Nets to the B-method, which supports the refinement and implementation of these systems. Nonetheless, this transformation does not include the implementation of the existing relay-based systems as computer-based ones, since it operates in a higher level of abstraction as a way to produce mixed systems that are referred as relay-based computer-controlled RIS. The use of a tool to support the implementation of the formal specification is also proposed in [Banci, Fantechi, and Gnesi 2004]. In this

work, the I-Logix statemate tool [Bienmüller, Damm, and Wittke 2000][Damm and Klose 2001] supports not only the description of the statecharts, but also the execution of the specification and automatic generation of source code based on the model.

Despite the fact that the works presented in this section aim at the implementation of the systems based on formal methodologies, not all of them present the refinement or code generation approaches. The logical models presented in [Roanes-Lozano et al. 2011], for instance, are implemented in the computer algebra system Maple [Char et al. 1983], whose Logic package and the language facilities support the sets notation required for such model implementation. Nonetheless, no refinement details have been presented in this work.

Although the formal specification of new computer-based RIS is desirable in order to produce safety proved systems, the formal specification of the existing safe-acknowledged relay-based technology as a way to transform them into computer-based systems has the potential to be beneficial to the industry. Some of the reasons that explain this fact are the exploitation of an existing logic instead of defining a new one, the preservation of the system behaviour and the improvement of the existing system safety-level. In the context of this present thesis, the formal specification of relay diagrams may allow not only its verification, but also its implementation through refinement towards computer systems, supporting the evolution of the existing relay-based technology to a computer-based one.

2.3.3 Other approaches

Some academical approaches for the specification and verification of local RIS installations have been proposed. These works are not related to the relay-based technology and do not aim at the implementation of these systems, but they allow the application of Formal Methods on the railway domain for the verification of RIS safety properties. Their interesting characteristics are worth to be mentioned as they share some contextual particularities with this present thesis.

An interesting work made in conjunction with railway engineers is presented in [James, Moller, et al. 2014]. It uses CSP||B [Schneider and Treharne 2005] in order to model the interlocking systems, combining event-based and state-based modelling. Although the creation of these models is based on track plans and control tables, this work proposes a decomposition of the railway network into smaller schemes, modelling and verifying smaller portions of the tracks (similar to our geographical context) and proposing a compositional verification approach. [James, Moller, et al. 2014] works with a different abstraction level when compared to our work, since the models are based on the trains movements on a railway network while we focus on the electrical circuits that control the track-side components. Nonetheless, [James, Moller, et al. 2014] has many similarities with our work that are worth to be mentioned: the models are developed together with industrial partners, it proposes the modelling and verification of small parts of a huge interlocking system in a compositional approach and it also proposes the transformation of a pre-existing domain-specific model into a formal specification. All these similarities present how similar works can be applied into two railway systems with different abstraction levels.

Z [Spivey and J. Abrial 1992] is a formal specification language with a strong basis in mathematical notations like Set Theory, Functions and Logic. It has been applied for the specification of Railway interlocking systems in many projects. In [N. A. Zafar 2006], [N. A.

Zafar 2009] and [N. A. Zafar, Khan, and Araki 2012], for instance, it is proposed the use of Graph Theory and Z in order to formally specify RIS and their safety properties in the interest of being able to prove the absence of collisions and derailments. Previously, a similar approach has been presented by the same author in [N. Zafar 2006] using the VDM-SL [Elmström, Larsen, and Lassen 1994] formal language and supporting tools. Z is also used in [Khan and N. A. Zafar 2009], which focuses on enforcing a safe distance between the trains. This work defines a length for the moving blocks based on the train velocity and it represents the tracks in a graph model. The system is factored into local train operation and global control operation and the safety is verified along a straight motion or along crossings. In [Janota 2000], it is presented a discussion about the use of Formal Methods in the railway domain and it presents a small example of the use of Z for the specification of a RIS. In this case, the specification is based on the train routes and the turnouts states. Although this work only presents a simple example of application of a formal language, it concludes that the lack of concurrency aspects on the specification may represent a limitation and it discusses the possibility of using CSP in order to complement the design.

In [Busard et al. 2015], it is proposed the creation of an executable model of a RIS in NuSMV based on the track layout, which is generally modelled in the SSI [Bellon 2014] language in the Belgian railways. So, this work proposes a translator from SSI to NuSMV and the generation of safety properties in order to verify the safety of these systems. Similarly to the context of this present thesis, this work proposes the formalisation of the systems described in an domain specific language, SSI.

Another work that proposes the specification of RIS based on the position of track-side components in a route is presented in [Hernando et al. 2012]. Differently from the other methodologies, this work proposes an algebraic approach that allows the verification of large systems, which is generally a limitation in this field. In this context, it presents an approach for representing Boolean formulae as polynomials and then logic problems are translated into algebraic problems. In a comparison with the time required for the system verification in other approaches, the algebraic model verification is extremely efficient and seems to be a promising method for detecting safety problems in large systems. This is an example of an alternative formal verification approach for the model checking, allowing the verification of large systems.

A completely different approach is presented in [Borälv 1998]. In this work, the formal verification of the system safety requirements is made using a tool that translates the system implementation into the input for the verification software. The differentiated approach for the specification, implementation and verification of the RIS in this work allows the use of formal proofs as a way to replace parts of the system-level test phase. The integration of formal methods in the development cycle of a RIS is also presented in [Cimatti et al. 1998a][Cimatti et al. 1998b]. This is one example that model checkers can still be successfully used in the RIS context. In this work, the safety logic of the part of the Ansaldo Italian interlocking system [Mongardi 1993] is specified in PROMELA, which is the input language of the SPIN model checker. This language has a process-based syntax that is based on CSP, allowing the definition of processes and the asynchronous messages they may exchange via channels. Indeed the specification presented in this work focuses on the information exchanged between the computerised RIS and the electrical components or the controllers. The integration of the formalisation and formal

verification steps with the current development process allows an advantageous cost/benefits relation. Furthermore it presents a successful example of the use of model checkers for the verification of complex RIS. This success is essentially due to the use of a powerful model checker and the specification of careful designed model.

2.4 Conclusions

The use of formal methodologies for the analysis, verification and safety assurance of the Railway Interlocking Systems is highly desired by industry. Thus, many different approaches for integrating formal specification methodologies in the RIS development have been presented in the literature. In this present thesis, we aim at the formalisation of the existing relay-based RIS structure and behaviour so it is possible to prove their safety and then evolve these systems to computer-based ones through refinement in a formal development methodology. So, our logical description of the system is based on the existing logic that is used for the development of these safety established systems.

After reviewing the literature about the RIS specification, it is possible to recognise some technologies that are commonly used. Graphs, for instance, are generally used to support the structural representation of the RIS track and stations layout. This data structure is a strong known abstraction that is supported by several studies and contains many standardised notations. Since it is a known data structure, it is easy to be understood by the different experts involved in the project. Based on the works presented in this chapter, one may conclude that the use of graphs in order to support the description of RIS logic is an approach well established in the RIS scientific community.

Regarding the verification of these systems, the use of model checkers is present in several works. Despite the limitation issues regarding the space state size reported in many works in the RIS functional level, there is not a significant discussion about it in the geographical level. The functional systems are related to a bigger scale of routing maps compared to geographical systems, which are more focused in local installations, so one may consider that this factor may contribute for having a smaller state space. However, this field still lacks a complete analysis of the use of model checkers in this context in order to understand and establish the limitations of this technology for the formal verification of geographical system specifications. Nevertheless, the creation of specification methodologies that support more than one verification approach can be an interesting solution in order to deal with model checking problems, as formal specification methodologies do not rely only on model checking. The comparison between the different verification approaches regarding efficiency may also be interesting in order to define the best technology for the verification of such systems. In this thesis, we aim at the creation of a model that can be used as basis for the formal specification of the RIS in different formal specification languages that support different verification approaches. As a consequence, this work is not limited to one technology and give an initial support for different companies use different tools according to their needs.

There is also a discussion in the literature about the best formal language for the formalisation and verification of such systems. Some languages that are successfully applied in determined contexts for the verification of specific properties are sometimes not helpful in other circumstances.

It is important to notice that each language has specific features, capabilities and particularities that make them more powerful in determined contexts. While languages focused on the system concurrency are strong for verifying the components synchronisation, languages based on simple Boolean logic are able to prove the system state safety based on simple expressions. The creation of an adaptable methodology that may support the specification of systems in different formal languages may be the key in order to assist the formalisation of different aspects of the same system and then strengthen their safety verification.

The industrial context that basis the formal specification is also a differentiating factor in order to define the details, priorities and objectives of the formal verification. The concept of safety in the Railway Interlocking Systems is generally similar, aiming at the avoidance of collisions and derailments. Nonetheless, the implementation of these systems are different in a way that the safety requirements are modelled in different manners. Regarding the relay-based systems, each company has its own design rules for modelling relay diagrams. Thus, it is a reasonable alternative to define contextually-specific approaches for the formalisation of these systems. Nonetheless, order to support the formal specification of wider range of relay-based RIS, the creation of a more adaptable general approach capable to be adjusted for different contexts tends to be academically and industrially interesting.

Based on the experiences and needs presented in the literature, this work aims at the formalisation of the relay-based RIS in a manner that it can be understood by the many experts involved, adapted to many different contexts and for the use of different specification languages. Besides, we focus on the complete transformation from the existing relay-based systems towards their computer-based implementation through a formal development approach. In this context, the formal specification proposed in our methodology can be used not only for the system verification but also as a first step towards their implementation. The next part of this present thesis presents the methodology for the formal specification and verification of such systems.

Part II

Methodology

Formalisation of Relay-based RIS: A Graph Approach

Contents

3.1 Introduction	66
3.2 RIS Basic Logical Description	67
3.2.1 Relay Diagrams Basic Structure	67
3.2.2 Electrical Components Structural and Behavioural Formalisation .	70
3.2.3 Timed Blocks Impact on the System State Definition	81
3.2.4 Capacitors and their Impact on the System State Definition	86
3.3 Flashing Lights: An Energy Source Variation	94
3.4 Formalisation Support for the System Verification	97
3.4.1 Structural Well-definedness Verification	97
3.4.2 Behavioural Safety Conditions Definition	100
3.5 Case Study Specification and Analysis	102
3.5.1 Structural Formalisation	102
3.5.2 Behavioural Formalisation and Verification	109
3.6 Discussion	113
3.7 Formal Specification Based on the Formalisation	114

3.1 Introduction

Relay diagrams are generally used by railway infrastructure managers for describing the relay-based Railway Interlocking Systems, modelling the electrical connection between the physical components. In this context, every structural or behavioural analysis of these diagrams are made by manual inspection, which tends to be error prone. An approach for formally modelling and verifying these systems may benefit the industry by allowing the proof of the system safety and correctness. However, before formally specifying the relay-based RIS it is important to understand all their structural and behavioural details.

So, instead of only formally specifying the relay-based RIS with the use of a formal specification language, this work proposes the analysis and basic mathematical formalisation of the relay-based RIS structure and behaviour. The main motivation of making this formalisation is the definition of a middle course between the informal models and the formal specification that may support the model transformation by providing a common and well established understanding of the system. Some of the major benefits of this formalisation are: the definition of a mathematical model that can be understood by railway engineers and formal specification experts, the possibility of proving safety properties of the system without using a specific formal specification language syntax, the creation of a complete mathematical model that can be easily translated to many different formal specification languages due to the use of common mathematical basis and the definition of a middle course model for the transformation of the existing relay diagrams to a formal specification, providing a mapping between the models syntax and behavioural logic.

This formalisation of the relay-based RIS is grounded on the relay-diagrams provided by SNCF and the concepts about their functioning as described in [Rétiveau 1987], [Schön et al. 2013] and [Schön et al. 2014], which are some of the most important references about the French Railway Interlocking Systems. In order to represent the system structure as described in the relay diagrams, a graph structure is defined based on Set Theory and Relations. In accordance with the knowledge about the behaviour of each electrical component and using the mathematical structural representation as support, it is possible to mathematically represent the behaviour of the system using Logic. In this context, it is possible to describe the precondition for achieving a determined electrical component state based on the state of the others. This behavioural representation is based on the known behaviour of each electrical component as described in [Rétiveau 1987], for instance, and it can be used as basis for the system formal specification and behavioural verification or validation, as presented in Chapter 4.

The structural representation presented in this chapter defines the basic structures that are needed in order to formalise the information contained inside the relay diagrams. This structural model may then be specialised in order to represent a specific system by adding the relay diagram specific information. The behavioural model is based on the structures defined in the structural model, so it is independent of the specific system information. This is because the behavioural model defines the possible states of each type of component, which is immutable. Thus, the description of a specific system is formed by the specialisation of the structural model together with the behavioural model as it is. The process of formalising a system based on the behavioural and structural formalisation presented in this chapter is depicted in Figure 3.1.

It is important to mention that the structural and behavioural formalisation presented in

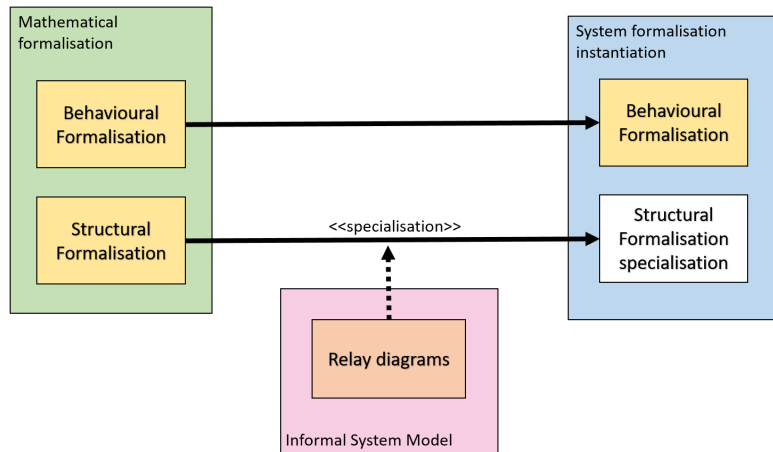


Figure 3.1 – Description of a specific system based on the RIS general structural and behavioural formalisation.

this chapter are based on the relay diagrams used in the SNCF French context. Nonetheless, this formalisation can also be modified and extended with the objective of supporting the specification of systems under other contexts. The main proposition in this chapter is the use of well mathematically grounded basis as a way to allow the modelling and verification of relay-based Railway Interlocking Systems. As the formalisation is grounded on the logical description of each component behaviour, it is possible to modify and extend this logic in order to consider different components and different behaviours.

3.2 RIS Basic Logical Description

As electrical systems, the RIS behaviour is completely dependent on the structural configuration of the wires and electrical components. Thus, as a way to model the system structure and support the behavioural formalisation, this section describes an approach for representing the system in a graph format based on how the electrical components are connected by wires. Then, the structural relation between the components is used as basis for the definition of their behavioural relation in the behavioural model. So, the relay-based RIS formalisation is divided into two types of definition: structural and behavioural. The former describes static information about how the electrical components are connected to others, which means that this information must never change during the system execution. The latter describes the states of the system based on the state of each component, an information that constantly changes during the system execution.

3.2.1 Relay Diagrams Basic Structure

A relay diagram presents all the connections between components in a graph-like format. By depicting how the electrical components are connected by wires, a relay diagram is a structural model that reflects the system implementation configuration. This structural model is essentially important in order to support the deduction of the system behaviour as the components electrification depends on their electrical connections. In order to formally analyse these systems, it is possible to mathematically represent this structure in a graph format based on Set Theory,

Logic and Relations, as presented in the Structural Definition SD1. These mathematical foundations provide a strong basis for performing mathematical proofs, as discussed in Sections 3.4.1 and 3.4.2.

Structural Definition SD1 (Relay Diagrams Graph Structure): A relay diagram can be represented by a graph where a set *Components* of nodes are connected by elements of the set *Wires* of edges according to the *Incidence_→* function. This graph may be mathematically represented by the following structures:

- a set *Components* of electrical components,
- a set *Wires* of electrical wires and
- a function $Incidence_{\rightarrow} \in (Wires \rightarrow (Components \times Components))$ from the set of wires to pairs of components, denoting the graph incidence function.

By describing the wired connection between the components in an incidence function, it states that a wire only connects two components. Nonetheless, as described in Chapter 1, junctions may be used in order to intermediate the connection between more than two components. In this context, it is important to notice that the model presented in this chapter for the description of the system structure is very permissive as it allows the definition of components relations that one may consider erroneous regarding the electrical circuits design. For instance, the graph does not limit the number of wires that can be connected to a component. This is a modelling decision in order to maintain the extensibility of the approach so it can be applied to different contexts with different modelling rules. Nonetheless, as a way to enforce the system design rules, this work proposes the definition of structural well-definedness properties, as presented later in Section 3.4.1, which can be used in order to verify the consistency of the definitions against the expected structure.

As the different types of electrical components are differentiated by their drawings inside relay diagrams, in this mathematical representation they must also be differentiated, which can be made in the form of components subsets. The components are divided into positive energy sources, negative energy sources, buttons, levers, lever contacts, monostable contacts, bistable contacts, junctions, capacitors, capacitors plates, resistors, blocks, outputs, monostable relays, bistable relays and bistable relays coils; which are represented in this model, respectively, in the form of the *Components* subsets *PosSources*, *NegSources*, *Buttons*, *Levers*, *LeverContacts*, *MonostableContacts*, *BistableContacts*, *Junctions*, *Capacitors*, *CapPlates*, *Resistors*, *Blocks*, *Outputs*, *MonostableRelays*, *BistableRelays* and *BistableCoils*. Each one of these sets is a subset of the *Components* set, as presented in the Structural Definition SD2.

Structural Definition SD2 (Components Differentiation): The RIS components are divided into the following *Components* subsets: *PosSources*, *NegSources*, *Buttons*, *Levers*, *LeverContacts*, *MonostableContacts*, *BistableContacts*, *Junctions*, *Capacitors*, *CapPlates*, *Resistors*, *Blocks*, *Outputs*, *MonostableRelays*, *BistableRelays* and *BistableCoils*. In this mathematical model, the component

differentiation may be denoted as:

$$\begin{aligned}
& PosSources \subseteq Components \quad \wedge \\
& NegSources \subseteq Components \quad \wedge \\
& Buttons \subseteq Components \quad \wedge \\
& Levers \subseteq Components \quad \wedge \\
& LeverContacts \subseteq Components \quad \wedge \\
& MonostableContacts \subseteq Components \quad \wedge \\
& BistableContacts \subseteq Components \quad \wedge \\
& Junctions \subseteq Components \quad \wedge \\
& Capacitors \subseteq Components \quad \wedge \\
& CapPlates \subseteq Components \quad \wedge \\
& Resistors \subseteq Components \quad \wedge \\
& Blocks \subseteq Components \quad \wedge \\
& Outputs \subseteq Components \quad \wedge \\
& MonostableRelays \subseteq Components \quad \wedge \\
& BistableRelays \subseteq Components \quad \wedge \\
& BistableCoils \subseteq Components.
\end{aligned}$$

It is important to note that some components may be composed by other components (sub-components). This is the case of bistable relays, levers, and capacitors which are composed by bistable relay coils, lever contacts and capacitors plates, respectively. These subcomponents have a semi-independent behaviour and they are independently connected to one or two different wires, i.e., the electricity does not flow between two subcomponents from the same component.

In relay diagrams, each component must have only one type. Furthermore, every component must have a defined type, since their behaviours must be known. So, in order to guarantee that the components sets are well defined, a well-definedness property may be described. These well-definedness properties are important in order to guarantee that the system structure is well defined according to the relay diagrams design rules.

In the context of the components differentiation, for instance, the system specification must guarantee that every component has one, and only one, known type. Thus, one must guarantee that the *Components* subsets are mutually disjoint and their union forms the *Components* set. This may be denoted based on the consideration that all these subsets are part of the family of sets *ComponentsSubsets*, which is defined as:

$$\begin{aligned}
ComponentsSubsets = \{ & PosSources, NegSources, Buttons, Levers, LeverContacts, \\
& MonostableContacts, BistableContacts, Junctions, Capacitors, \\
& CapPlates, Resistors, Blocks, Outputs, MonostableRelays, \\
& BistableRelays, BistableCoils \}.
\end{aligned}$$

So, based on the *ComponentsSubsets*, it is possible to define that:

$$\text{Components} = \bigcup \text{ComponentsSubsets} \wedge$$

$$\forall S1, S2. ((S1 \in \text{ComponentsSubsets} \wedge \\ S2 \in \text{ComponentsSubsets} \wedge \\ S1 \neq S2) \Rightarrow S1 \cap S2 = \emptyset).$$

A more detailed discussion about the well-definedness properties is presented in Section 3.4.1.

The definitions introduced in this section represent the most basic structure of a relay diagram: the components connections and differentiation. This plays an important part on mathematically describing and analysing the relay-based Railway Interlocking Systems. The behavioural definition of these diagrams are generally deduced based on the known behaviour of each component. So, in this formalisation, it is possible to define the behavioural logic of the system based on the components behaviour and relations as well.

3.2.2 Electrical Components Structural and Behavioural Formalisation

The behaviour of each electrical component is important for the definition of the complete system behaviour. While some components execute functions when electrified, others are responsible for controlling the flux of electrical current as a way to activate/deactivate components. This electrical flux control is based on the fact that the connection between the components is not fixed. Some components assume states during the execution of the system that restrict their connections as a way to block the flow of electrical current inside the wires. The components whose connections with wires may be cut are: contacts, buttons and levers.

Contacts may be monostable or bistable, according to the relay they are related to. They are generally able to assume two different states according to their physical positions. A monostable contact may assume the states *up* or *down*, while a bistable contact may assume the states *right* or *left*, as presented in Figure 3.2. In each state a contact may be providing and/or blocking connections between other components, affecting their electrification. The contacts states are part of the system behavioural definition, so, the Behavioural Definitions BD1 and BD2, present, respectively, how the state of monostable and bistable contacts may be logically represented.

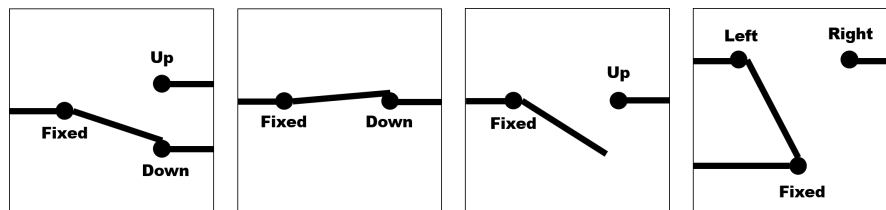


Figure 3.2 – Possible contacts positions in a relay diagram.

In order to be able to define the contacts states representation, one must consider the existence of constant universal sets, which define the possible states that these components may assume. The Boolean set \mathbb{B} is one example of a constant universal set. Other sets may be specifically

defined for components that may assume specific states other than the Boolean values.

Behavioural Definition BD1 (Monostable Contacts States Representation): Monostable contacts may assume the states *up* or *down*. The state of every monostable contact of a relay diagram may be represented by a total function. So, given the constant universal set U_D :

$$U_D = \{up, down\},$$

the monostable contacts states may be represented by a total function from the monostable contacts sets to U_D , which may be mathematically represented as:

$$MonoContactsSt_{\rightarrow} \in (MonostableContacts \rightarrow U_D)$$

Behavioural Definition BD2 (Bistable Contacts States Representation): Bistable contacts may assume the states *right* or *left*. The state of every Bistable contact of a relay diagram may be represented by a total function. So, given the constant universal set R_L :

$$R_L = \{right, left\},$$

the Bistable contacts states may be represented by a total function from the bistable contacts sets to R_L , which may be mathematically represented as:

$$BistContactsSt_{\rightarrow} \in (BistableContacts \rightarrow R_L).$$

As total functions, these definitions guarantee that a contact may never assume two different states in the same system state as transient states are not modelled. Together with the inputs, these components may control the passage of electrical current inside the wires by blocking or allowing the current to flow in determined wires.

The system inputs are the connection between the system and the environment. As these components states are controlled by the environment, they are responsible for causing the system internal state to change. Together with the components with timed behaviours (discussed in Section 3.2.3), they are responsible for the beginning of every chain reaction of the system. In this study, we consider the existence of three types of inputs: buttons, levers and contacts related to external relays.

Buttons have a simple behaviour, since they represent a contact that may be closed or opened when physically pressed. In order to avoid misunderstandings, we express the state of a button as a Boolean value, i.e., *true* or *false*. When a button is on its *true* state, it is allowing the current to flow inside the wires it is related to. Otherwise, when the button assumes the *false* state, it blocks the electrical current flow. There is no internal event that changes the button state, moreover it may be changed at any time by the environment. The state of each button may be described as a total function from the set of buttons to a set of Boolean values \mathbb{B} , as presented in the Behavioural Definition BD3.

Behavioural Definition BD3 (Buttons States Representation): The buttons states may be represented inside a total function from *Buttons* to the Boolean values *true* and *false*, where *true* represents that the button is closed and *false* represents that the button is opened. So, given the

constant universal set \mathbb{B} :

$$\mathbb{B} = \{true, false\},$$

it is possible to define a function $ButtonsSt$, such that:

$$ButtonsSt_{\rightarrow} \in (Buttons \rightarrow \mathbb{B}).$$

As a function, no button may have two states at the same system state. Furthermore, as a total function, every button must assume a state.

Unlike the buttons state definition, levers have a complex state caused by its complex structure. A lever controls two or more contacts that block or allow the current to flow in different wires. In this case, a lever may have two configurations that are named as $config_a$ and $config_b$ in this work. Besides, when a lever changes its configuration, all its related contacts states must change together. Figure 3.3 presents an example of this relation between levers and lever contacts. In this example, when the lever is in the $config_b$ configuration, only the contact $c3$ is closed, allowing the current to flow in the wires related to it. However, when this same lever has its configuration changed to $config_a$, all the lever contacts states change, opening the contact $c3$ and closing the contacts $c1$ and $c2$. Thus, the contacts configuration must be considered when expressing the possible levers states in our model. Before defining the levers states, the lever structure must be defined, as presented in the Structural Definition SD3.

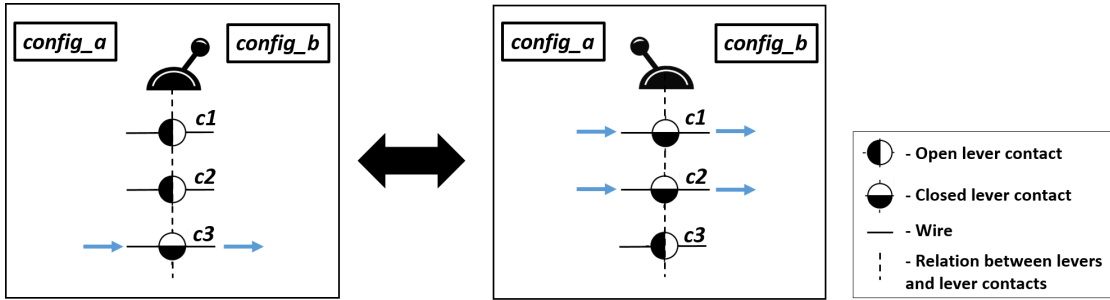


Figure 3.3 – Levers states transition example where the small arrows indicate where the current is flowing.

Structural Definition SD3 (Levers Structure Definition): The lever structure may be defined by a total function $LeverContactsRel_{\rightarrow}$, relating every lever contact to its respectively lever, and a relation $ConfigRel_{\leftrightarrow}$, which describes the association between the lever configuration and the contacts configuration. For this purpose, this latter relation must then link pairs of levers and their configurations with pairs of contacts and their configurations. Similarly to buttons, lever contacts may be opened or closed, which may be described by means of Boolean values (\mathbb{B}). So, given the constant universal set \mathbb{B} defined before and the set $LeverConfig$ of possible lever configurations:

$$LeverConfig = \{config_a, config_b\},$$

the function $LeverContactsRel_{\rightarrow}$ and the relation $ConfigRel_{\leftrightarrow}$ may be mathematically repre-

sented as:

$$\begin{aligned} LeverContactsRel_{\rightarrow} &\in (LeverContacts \rightarrow Levers) \wedge \\ ConfigRel_{\leftrightarrow} &\in ((Levers \times LeverConfig) \leftrightarrow (LeverContacts \times \mathbb{B})). \end{aligned}$$

By defining $LeverContactsRel_{\rightarrow}$ as a total function, each lever contact must be related to one, and only one, lever. This Structural Definition may then be used as basis in order to represent the lever states, as presented in the Behavioural Definition BD4.

Behavioural Definition BD4 (Levers States Representation): The lever states may be represented by the function $LeverSt_{\rightarrow}$, which defines the state of each lever, and the function $LeverContactsSt_{\rightarrow}$, which defines the state of each lever contact. So, given the constant universal sets \mathbb{B} and $LeverConfig$ defined before, one may define the functions $LeverSt_{\rightarrow}$ and $LeverContactsSt_{\rightarrow}$ as:

$$\begin{aligned} LeverSt_{\rightarrow} &\in (Levers \rightarrow LeverConfig) \wedge \\ LeverContactsSt_{\rightarrow} &\in (LeverContacts \rightarrow \mathbb{B}). \end{aligned}$$

As these components states are defined as total functions, it guarantees that every lever and lever contact always assume one, and only one, state. Considering the Structural Definition SD3, the levers and lever contacts states must meet the structural relation between these components. Thus, $LeverContactsSt_{\rightarrow}$ must be defined as:

$$LeverContactsSt_{\rightarrow} = ConfigRel_{\leftrightarrow}[LeverSt_{\rightarrow}]$$

Regarding the contacts related to external relays, their states may be described in the same way that other contacts. Indeed, these contacts are included in the Behavioural Definitions BD1 and BD2, since they are bistable and monostable contacts independently if the relay is modelled inside or outside the relay diagram. Nevertheless, since the behaviour of these components are entirely controlled by the environment (considering relays outside the diagram as part of the environment), they are not impacted by the system state. In fact, as inputs, these components are responsible for provoking the system state change by allowing or blocking the components electrification.

In order to define how the current flows inside the wires, it is important to differentiate two concepts: potential and real connections. In its most basic definition, the connection between two components is the electrical relation between them mediated by wires. In this case, the graph presented in the Structural Definition SD1 represents the potential connections between components as it is presented in the relay diagram, independently from their states. The real connections between components must consider the state of components, i.e., the physical connections in a specific state of the system. These connections depend on the buttons, levers and contacts states, which may block the flow of electrical current by cutting the connections. Thus, the real connections may be represented by the potential connections (incidence function of the graph) excluding the connections that do not exist in a specific state.

In order to define the real connections of a system in a specific state, it is important to define its disconnections, which is comprised by all edges whose connections to buttons, levers and

contacts are not achieved. However, in order to list the contacts disconnections, it is important to know what are the edges related to these contacts. In a determined state, a contact may provide the connection between two different wires. However, in this same state, the contact may also block a connection, as it is presented in Figure 3.4. In this figure, while the contact $C1$ allows the current to flow between the wires $W1$ and $W3$, it also blocks the current flow between $W1$ and $W2$. The determination of which wires are connected to the contacts in each of its states is essential in order to define the wires that are not connected in the system state. The contacts connections are presented in the Structural Definitions SD4 and SD5.

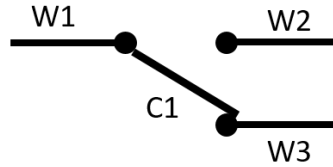


Figure 3.4 – Representation of a contact and its connections.

Structural Definition SD4 (Monostable Contacts Connections): The monostable contacts connections may be represented by a function from the pairs of monostable contacts and their states to edges, which indicate in which contact state the contact is connected to which edge. So, given the constant universal set U_D defined before, the monostable contacts connections may be mathematically represented as:

$$MonoContactsConn_{\rightarrow} \in ((MonostableContacts \times U_D) \rightarrow Wires).$$

Structural Definition SD5 (Bistable Contacts Connections): The bistable contacts connections may be represented by a function from the pairs of bistable contacts and their states to edges, which indicate in which contact state the contact is connected to which edge. So, given the constant universal set R_L defined before, the bistable contacts connections may be mathematically represented as:

$$BistContactsConn_{\rightarrow} \in ((BistableContacts \times R_L) \rightarrow Wires).$$

Regarding these definitions it is important to notice that the relation between the contacts positions and the connected wires is a partial function. As presented in Figure 3.2, a contact may not be connected to a wire on one of its positions. Thus, the use of a partial function in this context is the best solution in order to guarantee that contacts can be disconnected during the system execution.

Based on the contacts connections definitions, it is possible to define the system disconnections, which is a list of every edge (wire) that is not connected in its both extremities to electrical components. In this situation, a system disconnection is the edge which has one of its extremity completely physically disconnected due to a button, lever or contact state, blocking the current flow. In order to list the system disconnections, one must identify which connections from the graph incidence function are not achieved due to the contacts, buttons and lever states. The system disconnections are presented in the Behavioural Definition BD5.

Behavioural Definition BD5 (System Disconnections): The system disconnections is represented by all the edges related to contacts, buttons or lever contacts in the graph incidence function, which are not connected in the determined system state. This may be mathematically represented as:

Disconnections =

$$(\text{ran}(\text{MonoContactsSt}_{\rightarrow} \triangleleft \text{MonoContactsConn}_{\leftrightarrow}))$$

∪

$$\text{ran}(\text{BistContactsSt}_{\rightarrow} \triangleleft \text{BistContactsConn}_{\leftrightarrow})$$

∪

$$(\text{Incidence}_{\rightarrow}^{-1}[\text{ran}(\text{Incidence}_{\rightarrow}) \triangleright (\text{LeverContactsSt}_{\rightarrow}^{-1}[\{\text{false}\}] \cup \text{ButtonsSt}_{\rightarrow}^{-1}[\{\text{false}\}])])$$

∪

$$(\text{Incidence}_{\rightarrow}^{-1}[(\text{LeverContactsSt}_{\rightarrow}^{-1}[\{\text{false}\}] \cup \text{ButtonsSt}_{\rightarrow}^{-1}[\{\text{false}\}]) \triangleleft \text{ran}(\text{Incidence}_{\rightarrow})]).$$

So, the system disconnections is formed by the union of:

- The set of wires that cannot be connected to the monostable contacts due to the contacts states;
- The set of wires that cannot be connected to the bistable contacts due to the contacts states;
- The set of wires from the incidence function that connect components to opened buttons or lever contacts, i.e., the edge of each incidence whose second component is an opened button or an opened lever contact;
- The set of wires from the incidence function that connect opened buttons or lever contacts to other components, i.e., the edge of each incidence whose first component is an opened button or an opened lever contact.

Based on the system disconnections, one may define what are the system real connections, which is constituted by the graph incidence function without the disconnected edges. The system real connections represent the physical connections of the components during the system execution, which is extremely important in order to define the components electrification conditions.

Behavioural Definition BD6 (System Real Connections): The system real connections is defined by the connections presented in the relay diagram graph without the edges that are not physically connected (disconnections). This may be mathematically represented by the function RC_{\leftrightarrow} , which represents the incidence function where disconnected wires are removed:

$$RC_{\leftrightarrow} = (\text{Disconnections} \triangleleft \text{Incidence}_{\rightarrow}).$$

The concept of real connections and its importance for the system state definition can be better explained using the circuit example presented in Figure 3.5, which is part of the ITCS

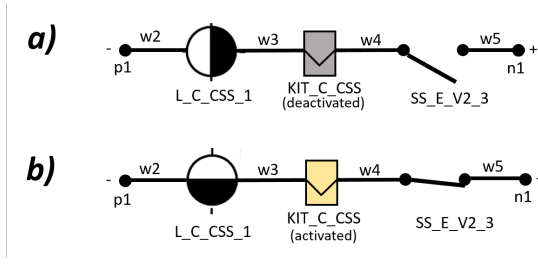


Figure 3.5 – Impact of the components states over the system state.

case study. The incidence function of this circuit is:

$$Incidence_{\rightarrow} = \{(w2, (p1, L_C_CSS)), (w3, (L_C_CSS_1, KIT_C_CSS_1)), \\ (w4, (KIT_C_CSS, SS_E_V2_3)), (w5, (SS_E_V2_3, n1))\}.$$

According to the graph structural definition, there is always a connection between KIT_C_CSS and the energy sources. On the other hand, the system real connections depend on the components states, which are represented as the circuits *a* and *b* in the Figure 3.5. In this same example, when $L_C_CSS_1$ is open and $SS_E_V2_3$ is in the *down* state (circuit *a*), the system real connections function considers only the connection provided by the wire $w4$:

$$RC_{\leftrightarrow} = \{(w4, (KIT_C_CSS, SS_E_V2_3))\}.$$

By disconsidering the components disconnections in the RC_{\leftrightarrow} definition, it is possible to establish a condition for determining whether components, like the relay KIT_C_CSS , are activated or not.

On its most basic definition, a component is electrified if there is a path between a positive and a negative energy sources that contains this component. In this context, a path is defined as a sequence of connected nodes, as presented in the Section 1.3.4. This electrification definition must not be generalised, as it does not consider the impact of timed components. Besides, this definition holds for non-timed components that can be activated and which have only two connections, i.e., monostable relays, bistable relays coils and outputs. The Behavioural Definition BD7 mathematically describes the basic components electrification condition. The use of timed components and a more general electrification condition are defined later in the Sections 3.2.3 and 3.2.4.

Behavioural Definition BD7 (Basic Components Electrification Condition): A component *comp* from the set of monostable relays, bistable relays coils or outputs, is electrified in a determined state if there is a path from a positive to a negative energy source that contains the component *comp*. This definition holds in the case where no timed component is used. This condition may be logically expressed as:

$$electrified(comp) \Leftrightarrow \exists ps, ns, Pa_{\rightarrow}. (ps \in PosSources \wedge \\ ns \in NegSources \wedge Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge \\ comp \in ran(Pa_{\rightarrow})).$$

Based on this definition, in Figure 3.5, one may consider that the relay is activated if there exists a path in the RC_{\leftrightarrow} function between the positive and negative energy sources $p1$ and $n1$, respectively. In this context, the path $\langle p1, L_C_CSS_1, KIT_C_CSS, SS_E_V2_3, n1 \rangle$ begins in the positive energy source and finishes in a negative one, moreover, it is a path in the RC_{\leftrightarrow} that contains the relay. In this example, this path exists only in the circuit b of the Figure 3.5, i.e., if the lever contact $L_C_CSS_1$ and the contact $SS_E_V2_3$ are not disconnected.

The component electrification plays the most important part on the execution of the system, since it is responsible for the component activation. Roughly speaking, the majority of the electrical components that are energy dependent are activated when they are electrified. Timed components are the only one exception in this case. Although the word "activation" is usually used as a synonym of "electrification", they do not mean the same thing, as some components may be electrified and take some stipulated time to be activated. Relays are examples of components that are activated when they are electrified. The behaviour of this component plays an important part on the system logic, since they are the main control of the electric current flow inside the electrical circuits

In relay diagrams, relays have a major importance: once activated, they change the contacts states, which may activate or deactivate other relays creating a chain effect until the system reaches a stationary state. While monostable relays have a simple structure containing only one coil, bistable relays have a doubled coiled structure that adds complexity to its structural and behavioural representation. Before defining the relays states representation, it is important to understand the structure of bistable relays, which is represented in the Structural Definition SD6. In order to specify the bistable relay states, their electromagnetic coils must be placed in the right and left positions. Furthermore, there must exist a relation between the bistable relays and their coils, so their states may be linked.

Structural Definition SD6 (Bistable Relays Structural Definition): Each bistable relay is composed by two coils associated with fixed positions (right and left). The relation between bistable relays and their coils may be represented by a total function $BistableCoilsRel_{\rightarrow}$ from $BistableCoils$ to $BistableRelays$. Furthermore, the position of each coil inside the relay may be described by a total function $CoilSide_{\rightarrow}$ from $BistableCoils$ to the positions defined in the set R_L .

So, given the constant universal set R_L defined before, The functions $BistableCoilsRel_{\rightarrow}$ and $CoilSide_{\rightarrow}$ may be defined, such that:

$$\begin{aligned} BistableCoilsRel_{\rightarrow} &\in (BistableCoils \rightarrow BistableRelays) \wedge \\ CoilSide_{\rightarrow} &\in (BistableCoils \rightarrow R_L). \end{aligned}$$

By defining these relations as total functions, it guarantees that every coil is related to one and only one bistable relay. Besides, it guarantees that every bistable relay coil has one side defined. However, this structural definition allows bistable relays to have more than two coils, which is not a possible configuration. Thus, in order to guarantee the system well-definedness, one must establish that each bistable relay must have only two coils, one on the left side and other

on the right side. This property can be mathematically described as:

$$\begin{aligned} \forall br.(br \in BistableRelays \Rightarrow \exists c1, c2.(c1 \in BistableCoils \wedge c2 \in BistableCoils \wedge \\ BistableCoilsRel_{\rightarrow}^{-1}[\{br\}] = \{c1, c2\} \wedge \\ CoilSide_{\rightarrow}(c1) = right \wedge \\ CoilSide_{\rightarrow}(c2) = left \wedge)) \end{aligned}$$

The state of a component regarding its activation may be defined as a Boolean value, where *true* represents the activated state and *false* represents the deactivated state. Concerning relays, their states depend on the activation and deactivation of their electromagnetic coils. Thus, a monostable relay may assume the activated or deactivated states, since they contain only one coil. However, due to the double coil structure of bistable relays, their state definition is also more complex. Nonetheless, their states may be simply represented based on their impact over the system, i.e., by the states of their related contacts. So, one may consider that a bistable relay may assume the states *right* or *left*. The state representation of monostable and bistable relays are presented in the Behavioural Definitions BD8 and BD9, respectively.

Behavioural Definition BD8 (Monostable Relays States Representation): As monostable relays may be activated or deactivated, their state may be represented by a total function from the set of monostable relays to the set of Boolean values \mathbb{B} , where *true* represents the activated state and deactivated otherwise. So, given the constant universal set \mathbb{B} of boolean values defined before, one may mathematically represent the monostable relays states by the function $MonoRelaysSt_{\rightarrow}$, such that:

$$MonoRelaysSt_{\rightarrow} \in (MonostableRelays \rightarrow \mathbb{B}).$$

Behavioural Definition BD9 (Bistable Relays States Representation): As bistable relays may assume the states *right* or *left*, the state of these components may be represented by a total function from the set of bistable relays to R_L . So, given the constant universal set R_L defined before, one may mathematically represent the bistable relays states by the function $BistRelaysSt_{\rightarrow}$, such that:

$$BistRelaysSt_{\rightarrow} \in (BistableRelays \rightarrow R_L).$$

Based on the relays state representation and the components electrification condition, one may define whether each relay assumes each configuration in a determined state. This relay state definitions is presented in the Behavioural Definitions BD10 and BD11. A monostable relay is a simple component, so it is activated once electrified and deactivated otherwise. However, due to the bistable relays complex structure, the definition of their states must take into consideration all the possible configurations of both electromagnetic coils.

Behavioural Definition BD10 (Monostable Relays States Definition): Monostable relays are activated once they are electrified, and deactivated when they are no longer electrified. So, given the constant universal set of Boolean values \mathbb{B} , the state of each monostable relay in a determined

system state may be defined as:

$$\forall mr. ((mr \in \text{MonostableRelays}) \Rightarrow (\text{MonoRelaysSt}_{\rightarrow}(mr) \Leftrightarrow \text{electrified}(mr))).$$

Behavioural Definition BD11 (Bistable Relays States Definition): The states of bistable relays are defined by the different possibilities of activation of their coils according to the following conditions:

- if only the right coil is activated, the relay assumes the *right* state;
- if only the left coil is activated, the relay assumes the *left* state;
- if both coils are activated or deactivated, the relay must continue assuming its previous state;

So, given the constant universal set of Boolean values \mathbb{B} and considering the existence of a function:

$$\text{RelayPrevSt}_{\rightarrow} \in (\text{BistableRelays} \rightarrow L_R)$$

that defines the last stable state of a bistable relay, the state of each bistable relay in a determined system state may be defined as:

$$\begin{aligned} &\forall br, c1, c2. ((br \in \text{BistableRelays} \wedge \\ &c1 \in \text{BistableCoils} \wedge c2 \in \text{BistableCoils} \wedge c1 \neq c2 \wedge \\ &\text{BistableCoilsRel}_{\rightarrow}^{-1}[\{br\}] = \{c1, c2\}) \Rightarrow \\ &(((\text{electrified}(c1) \wedge \neg \text{electrified}(c2)) \Rightarrow \text{BistRelaysSt}_{\rightarrow}(br) = \text{CoilSide}_{\rightarrow}(c1)) \wedge \\ &((\neg \text{electrified}(c1) \wedge \text{electrified}(c2)) \Rightarrow \text{BistRelaysSt}_{\rightarrow}(br) = \text{CoilSide}_{\rightarrow}(c2)) \wedge \\ &((\neg \text{electrified}(c1) \wedge \neg \text{electrified}(c2)) \Rightarrow \text{BistRelaysSt}_{\rightarrow}(br) = \text{RelayPrevSt}_{\rightarrow}(br)) \wedge \\ &((\text{electrified}(c1) \wedge \text{electrified}(c2)) \Rightarrow \text{BistRelaysSt}_{\rightarrow}(br) = \text{RelayPrevSt}_{\rightarrow}(br))). \end{aligned}$$

Although the consideration of a previous state may require an extra effort when analysing a specific system state, the safety guarantee must not be impacted by the bistable relay previous state. The reason for this is that the succession of events will only reach the states where both coils are activated/deactivated in order to change the relay state in a next step. In fact, these "continued" states generally do not create a dangerous state. Nonetheless, if the execution accuracy (the precise state evolution of this component) is required, the state succession may be considered in the system formalisation, as presented in the definition above.

The contacts states are completely dependent on the relays states. An activated monostable relay must have their contacts in the *up* state, otherwise the contacts fall to the *down* state. Bistable relays and their related contacts must always assume the same state. Before defining the contacts states, it is important to define the relation between relays and contacts, which is part of the system structure. This relation is presented in the relay diagrams as semi-dotted vertical lines. The structural relation between relays and contacts is presented in the Structural Definitions SD7 and SD8.

Structural Definition SD7 (Structural Relation Between Monostable Contacts and Relays): The relation between monostable contacts and relays may be represented by a total function

from the set containing all the system monostable contacts to a set containing all the system monostable relays:

$$\text{MonoRelayContactsRel}_{\rightarrow} \in (\text{MonostableContacts} \rightarrow \text{MonostableRelays}).$$

Structural Definition SD8 (Structural Relation Between Bistable Contacts and Relays): The relation between bistable contacts and relays may be represented by a total function from the set containing all the system bistable contacts to a set containing all the system bistable relays:

$$\text{BistRelayContactsRel}_{\rightarrow} \in (\text{BistableContacts} \rightarrow \text{BistableRelays}).$$

The use of total function guarantees that each contact is related to one, and only one, relay. As the contacts states are directly linked to the relays states, the contacts states may be defined based on the relation between relays and contacts, as presented in the Behavioural Definitions BD12 and BD13.

Behavioural Definition BD12 (Monostable Contacts States Definition): In a determined system state, a monostable contact related to a monostable relay must be in the *up* state when the relay is activated (true) and in the *down* state when the relay is deactivated (false). So, given the constant universal set U_D and \mathbb{B} defined before, the state of all the monostable contacts in a determined system state may be defined as:

$$\begin{aligned} \forall mc. (mc \in \text{MonostableContacts} \Rightarrow \\ ((\text{MonoRelaysSt}_{\rightarrow}(\text{MonoRelayContactsRel}_{\rightarrow}(mc)) \Rightarrow \text{MonoContactsSt}_{\rightarrow}(mc) = up) \wedge \\ (\neg \text{MonoRelaysSt}_{\rightarrow}(\text{MonoRelayContactsRel}_{\rightarrow}(mc)) \Rightarrow \text{MonoContactsSt}_{\rightarrow}(mc) = down))). \end{aligned}$$

Behavioural Definition BD13 (Bistable Contacts States Definition): Every related bistable contact and bistable relay must assume the same states in every system stationary state. So, given the constant universal set R_L , the state of all bistable contacts in a determined system state may be defined as:

$$\begin{aligned} \forall bc. (bc \in \text{BistableContacts} \Rightarrow \\ (\text{BistContactsSt}_{\rightarrow}(bc) = \text{BistRelaysSt}_{\rightarrow}(\text{BistRelayContactsRel}_{\rightarrow}(bc)))). \end{aligned}$$

In some cases, specific relays are not drawn in the relay diagram, which means that they are inputs of the system and their states are controlled by the environment. In this case, a contact related to an external relay is indirectly controlled by the environment. Nonetheless, all these components and their states are still normally defined in the formalisation inside their respective sets and functions.

Regarding the outputs activation, these components have no impact inside the system. They generally represent lamps, antennas or any type of signalisation that have impact over the environment. Their activation follows the general Component Electrification Definition (Behavioural Definition BD7), since they are activated once electrified. In this analysis, the activation of outputs only impacts on the safety verification of the complete system, which is presented later in this chapter. The Behavioural Definitions BD14 and BD15 present, respectively, the outputs

state representation and definition in this formalisation.

Behavioural Definition BD14 (Outputs States Representation): As outputs may be activated or deactivated, their states may be represented by a total function from the set of outputs to the set of Boolean values \mathbb{B} , where *true* represents the activated state and deactivated otherwise. So, given the constant universal set \mathbb{B} of Boolean values defined before, one may mathematically represent the outputs states by the function $OutputsSt_{\rightarrow}$, such that:

$$OutputsSt_{\rightarrow} \in (Outputs \rightarrow \mathbb{B}).$$

Behavioural Definition BD15 (Outputs States Definition): Outputs are activated once they are electrified, and deactivated when they are no longer electrified. So, given the constant universal set of Boolean values \mathbb{B} , the state of each output in a determined system state may be defined as:

$$\forall out.(out \in Outputs \Rightarrow (OutputsSt_{\rightarrow}(out) \Leftrightarrow electrified(out))).$$

Besides the basic electrification condition presented in the Behavioural Definition BD7, in some systems one must consider the existence of components that produce energy as a way to electrify other electrical components. The components that may produce energy are blocks and capacitors. They are responsible for giving a notion of time to the system as a way to retard the activation or deactivation of components. The specification of these components requires a mixed approach, by considering the external influence (time) over the component and its influence over the rest of the system.

3.2.3 Timed Blocks Impact on the System State Definition

There is still one aspect that has a major impact on the activation and deactivation of components: time. In this work, time is considered as an environmental influence over the system and some components may require a certain time to pass in order to be activated or deactivated. In this work, the only components that are time dependent are blocks and capacitors.

In the case of blocks, they may have their activation or deactivation delayed after a determined time. Thus, time is considered as an input and described as a Boolean value, indicating whether the block time has passed (*true*) or not (*false*). This modelling decision is explained by the fact that only the preconditions for the block activation are important for the system state definition and analysis. However, one must not ignore the importance of time for the guarantee of the system safety. In this work abstraction level, the logical time is useful in order to establish the components states preconditions. Nonetheless, in a later implementation of this logic in a formalism that supports the description of timed state successions, the physical time must be considered as it is important in order to guarantee that the trains and the electrical circuits have enough time to execute their functions. So, although time is treated here as an input, it may be later implemented by a clock in the system computer-based implementation. As an input, the passing of time may be described as presented in the Behavioural Definition BD16.

Behavioural Definition BD16 (Block Passing of Time State Definition): For each timed block, the time necessary for their activation or deactivation may be abstracted by a variable indicating

if the required time has passed or not. In this work the time is considered as an environmental aspect. So, given the constant universal set \mathbb{B} of Boolean values, the time state for each block, i.e., whether the time has passed or not, may be represented as a function from the set of blocks to the Boolean values, such that:

$$BlockPassOfTime_{\rightarrow} \in (Blocks \rightarrow \mathbb{B})$$

As a total function, it describes that every block has a state related to the passing of time. In this case, the *true* value indicates that the required time has passed and the *false* value indicates that this time has still not passed.

The block electrification and activation follow different rules which require different definitions. Structurally, a block may have one of two types: timed activation or timed deactivation block. A timed activation block is represented in a relay diagram with a thicker line on the top of its component drawing, as detailed in Chapter 1. This component has its activation delayed for a certain time after its electrification. On the other hand, a timed deactivation block, represented in the relay diagram with a thicker line on the bottom of its component, has its deactivation delayed for a certain time. A graphical representation of these blocks as they are used inside relay diagrams is presented in Figure 3.6. In order to represent the blocks behaviour, one must have them structurally differentiated. The blocks structural differentiation is presented in the Structural Definition SD9.

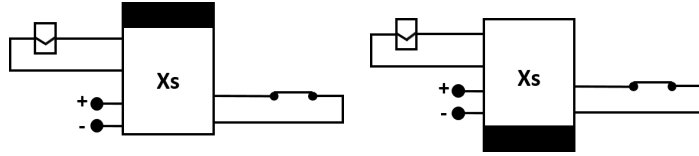


Figure 3.6 – A timed activation block (on the left) and a timed deactivation block (on the right).

Structural Definition SD9 (Blocks Differentiation): Blocks are differentiated into timed activation and timed deactivation Blocks. This differentiation may be mathematically represented by a function from the set of blocks to the set of block types. So given the constant universal set $BlockPossTypes$, such that:

$$BlockPossTypes = \{timed_act, timed_deact\},$$

which contains the block possible types *timed_act* (timed activation) and *timed_deact* (timed deactivation), it is possible to differentiate the blocks by the function $BlockTypes_{\rightarrow}$, defined below:

$$BlockTypes_{\rightarrow} \in (Blocks \rightarrow BlockPossTypes).$$

As a total function, it defines that every block must have one type. Furthermore, structurally speaking, one may consider that the blocks connections are differentiated into dependent and independent connections. In the relay diagram, the block is connected directly to a positive and a negative sources of energy. However, the block is electrified only if its independent connections are connected to each other or to other energy sources. These connections are

called as "independent" since they do not require the block activation in order to provide energy, i.e., the electricity flows in these contacts independently from the block state. On the other hand, the block dependent connections are the ones whose electricity flow is controlled by the block activation, i.e., once activated, the block provides energy to its dependent connections. In fact, there is no visual difference between the block dependent and independent connections inside the relay diagram, nevertheless, as their differentiation impacts on the block behavioural definition, they must be differentiated in this formalisation. The block connections differentiation is presented in the Structural Definition SD10.

Structural Definition SD10 (Blocks Connections Differentiation): The block connections may be differentiated into dependent and independent connections. This differentiation may be represented by two relations from the set of blocks to the set of components. The relation $BlockDepConn_{\leftrightarrow}$ and $BlockIndConn_{\leftrightarrow}$, representing the dependent and independent connections between the block and other components, respectively, may be mathematically represented as:

$$BlockDepConn_{\leftrightarrow} \in (Blocks \leftrightarrow Components) \wedge \\ BlockIndConn_{\leftrightarrow} \in (Blocks \leftrightarrow Components).$$

In this context, it is important to establish that the dependent and independent connections of each block are always different, which may be mathematically defined as:

$$BlockDepConn_{\leftrightarrow} \cap BlockIndConn_{\leftrightarrow} = \{\}.$$

The electrification of a block does not imply directly on its activation. So it is important to differentiate when the block is electrified or activated separately. The blocks states representation is presented in the Behavioural Definition BD17.

Behavioural Definition BD17 (Blocks States Representation): The blocks states may be represented by total functions from the set of blocks to the set of Boolean values, where the values *true* and *false* indicate if the block is activated/electrified or deactivated/not electrified, respectively. So, given the constant universal set \mathbb{B} defined before, the blocks states may be represented by the functions $BlockIsElectrified_{\rightarrow}$ and $BlockIsActivated_{\rightarrow}$, indicating whether the blocks are electrified and activated, respectively. These functions are mathematically represented as:

$$BlockIsElectrified_{\rightarrow} \in (Blocks \rightarrow \mathbb{B}) \wedge \\ BlockIsActivated_{\rightarrow} \in (Blocks \rightarrow \mathbb{B}).$$

As total functions, every block must have a state defined. The block electrification does not follow the component electrification condition presented in the Behavioural Definition BD7. A block is electrified by meeting one of the following possible conditions:

- the closing of a cycle that begins and finishes in the block through both its independent connections,
- the existence of a path between the block and a source of energy through an independent connection.

Once one of these conditions is satisfied, the electricity flows inside the block, whose activation, however, may depend on the passing of time. A timed activation block is activated only when it is electrified and the block time has passed. In the other hand, the timed deactivation block is deactivated only when the block is not electrified and the time has passed. The conditions for the activation and deactivation of blocks is presented in the Table 3.1. The block states definition is presented in the Behavioural Definition BD18.

Table 3.1 – Conditions that must be satisfied for the activation of blocks from each type.

Electrification	Passing of time	Timed Activation block	Timed deactivation block
Electrified	Not passed	deactivated	activated
Electrified	Passed	activated ^a	activated ^a
Not electrified	Not passed	deactivated	activated
Not electrified	Passed	deactivated ^b	deactivated

^aTime has no impact on the activation of this component, so this state is defined only by the block electrification state

^bTime has no impact on the deactivation of this component, so this state is defined only by the block electrification state

Behavioural Definition BD18 (Blocks States Definition): A block is electrified if there is a cycle that begins and finishes in the block passing through its independent connections or if there is a path between the block and an energy source that passes through an independent connection. A timed activation block is activated if the block is electrified and its specified time has passed, otherwise the block is deactivated. Furthermore, a timed deactivation block is deactivated if it is not electrified and its specified time has passed, otherwise it is activated. All these conditions may be generally mathematically defined for all the system blocks as:

$$\begin{aligned}
& \text{BlockIsElectrified}_{\rightarrow}(bl) \Leftrightarrow \\
& \exists es, Pa_{\rightarrow}. (es \in (PosSources \cup NegSources) \wedge \\
& ((Pa_{\rightarrow} \in \text{path}(bl, es, RC_{\rightarrow}) \wedge (bl, \text{first}(\text{tail}(Pa_{\rightarrow}))) \in \text{BlockIndConn}_{\leftrightarrow})) \vee \\
& (Pa_{\rightarrow} \in \text{cycle}(bl, RC_{\rightarrow}) \wedge ((bl, \text{first}(\text{tail}(Pa_{\rightarrow}))) \in \text{BlockIndConn}_{\leftrightarrow}) \wedge \\
& ((bl, \text{last}(\text{front}(Pa_{\rightarrow}))) \in \text{BlockIndConn}_{\leftrightarrow})))
\end{aligned}$$

$$\begin{aligned}
& \text{BlockIsActivated}_{\rightarrow}(bl) \Leftrightarrow \\
& ((\text{BlockTypes}_{\rightarrow}(bl) = \text{timed_act} \wedge \text{BlockIsElectrified}_{\rightarrow}(bl) \wedge \text{PassOfTime}_{\rightarrow}(bl)) \vee \\
& (\text{BlockTypes}_{\rightarrow}(bl) = \text{timed_deact} \wedge \text{BlockIsElectrified}_{\rightarrow}(bl)) \vee \\
& (\text{BlockTypes}_{\rightarrow}(bl) = \text{timed_deact} \wedge \neg \text{BlockIsElectrified}_{\rightarrow}(bl) \wedge \neg \text{PassOfTime}_{\rightarrow}(bl))).
\end{aligned}$$

Based on the block states definition, it is possible to extend the non-temporised components electrification condition in order to consider the possibility of the blocks to produce energy in their cycles. So, if a relay diagram contains blocks, one must consider that an electrical component is electrified if:

- There is a path between two different poles of energy sources that pass through the

component;

- There is a cycle that begins and finishes on a block that passes through the component and both independent connections of the block;
- There is a path between a block and an energy source that passes through the component and a block independent connection;
- There is a cycle that begins and finishes on an activated block that passes through the component and both dependent connections of the block.

These conditions are represented in Figure 3.7 respectively from left to right, depicting the four different manners a relay may be electrified when considering the existence of blocks. This new electrification condition may be defined as presented in the Behavioural Definition BD19. As well as the condition presented in the Behavioural Definition BD7, this block-related electrification condition also holds only for non-timed components that can be activated and which have only two connections, i.e., monostable relays, bistable relays coils and outputs.

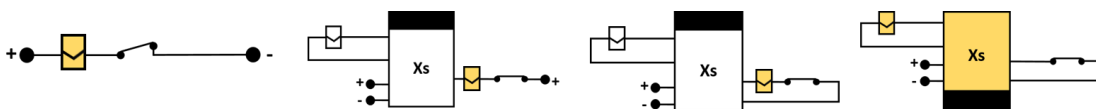


Figure 3.7 – Different manners to electrify a relay when considering the existence of blocks (the components highlighted in yellow are the ones that are activated).

Behavioural Definition BD19 (Block-related Electrification Condition): Non-temporised components are electrified when connected to: a positive and a negative energy sources; an energy source and a block independent connection; two block independent connections; or two block

dependent connections of an activated block. This may be mathematically expressed as:

$$\begin{aligned}
& electrified(comp) \Leftrightarrow \\
& ((\exists ps, ns, Pa_{\rightarrow}. (ps \in PosSources \wedge ns \in NegSources \wedge \\
& Pa_{\rightarrow} \in path(ps, ns, RC_{\rightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& (\exists bl, es, Pa_{\rightarrow}. (bl \in blocks \wedge \\
& es \in (PosSources \cup NegSources) \wedge \\
& Pa_{\rightarrow} \in path(bl, es, RC_{\rightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge \\
& (bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
& (\exists bl, Pa_{\rightarrow}. (bl \in blocks \wedge Pa_{\rightarrow} \in cycle(bl, RC_{\rightarrow}) \wedge \\
& comp \in ran(Pa_{\rightarrow}) \wedge \\
& (bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow} \wedge \\
& (bl, last(front(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
& (\exists bl, Pa_{\rightarrow}. (bl \in blocks \wedge Pa_{\rightarrow} \in cycle(bl, RC_{\rightarrow}) \wedge \\
& comp \in ran(Pa_{\rightarrow}) \wedge BlockIsActivated_{\rightarrow}(bl) \wedge \\
& (bl, first(tail(Pa_{\rightarrow}))) \in BlockDepConn_{\leftrightarrow} \wedge \\
& (bl, last(front(Pa_{\rightarrow}))) \in BlockDepConn_{\leftrightarrow})))
\end{aligned}$$

where the new conditions for the components electrification based on the use of blocks are highlighted in yellow.

Despite their complexity, the use of timed components is essential in order to preserve the system safety in determined cases. For instance, in order to change a signal from green to red, the system must wait a little time as a way to guarantee that any sensor that may have been delayed for any reason may indicate any possible presence in the tracks. This safety measure must be taken even in computer-based systems. Another way to force the system to wait a determined time is by the use of capacitors. However, these components have an extremely complex behaviour that must be carefully analysed, as presented in the next section.

3.2.4 Capacitors and their Impact on the System State Definition

Besides the Block, the other timed component that may be used inside the relay diagrams is the capacitor. This component has a dynamic behaviour that must be deeply analysed before its formalisation. The most important characteristics of this component are: the possibility of providing energy to the system, the double timed behaviour and the possibility of assuming charges on each of its side.

A capacitor may provide energy to the system when fully charged, but this component requires a certain time connected to the energy sources in order to be charged. Furthermore, a capacitor may only provide energy for a limited time. So, it is necessary to consider the two timed behaviours of this component in its formalisation. The charging time is essential in order to guarantee that the component is fully charged before providing energy. The discharging time

is important in order to establish the duration in which this component may act as an energy source.

Physically, a capacitor is formed by two conductive plates separated by a non-conductive region. When each plate of a capacitor is connected to a different pole of energy source, the energy cannot pass through the capacitor, but the energy sources cause each plate to store charges. The plate connected to the negative energy source stores negative charges, while the plate connected to the positive energy source stores positive charges. This process takes a specific time that depends on several conditions (like the material used, for instance). After storing a determined charge, a plate may produce energy by connecting it to another energy source with the opposite charge. As the plate is discharging, the energy can only be provided for a limited time.

In order to formalise these components, one must consider all the complexities inherent to the capacitors behaviour. However, there are some natural limitations about what can be modelled with the logic presented until now. The main problem is the definition of the capacitors states, which can only be made based on the capacitors previous states. This limitation is related to the following questions:

1. What are the charges of the plates of a charged capacitor that is not connected to the energy sources?
2. A capacitor that is connected to the positive and the negative energy sources is charging or discharging?

In order to answer the first question, it is necessary to consider what where the connections of the capacitor in one of the previous states. Similarly, the answer of the second question is related to the capacitor previous states, i.e., was it previously charged or discharged in the previous state? Are the charges connected to their opposite energy source poles? All these questions demonstrate how the definition of the capacitors state depends on their previous ones.

In the Behavioural Definition BD11, the description of the bistable relays states required the use of a previous state value, as this component state may be maintained indefinitely in determined circumstances. In the case of capacitors, however, the definition of previous states may be extremely complex and imprecise, since these components states may not be simply maintained over time, as they may charge, discharge and even change the plates charges during their execution. While a bistable relay may only switch the state to the other side (from *right* to *left* or from *left* to *right*), a disconnected capacitor has a list of possible future and previous states that are hard to be predicted and described in order to to define their exact state. In this context, while the bistable relays states may depend on their previous ones, the definition of the capacitors states may depend on many previous states. Thus, the complete definition of the capacitors states in basic logic may be unpractical without the use of a state succession logic as support.

With the purpose of limiting this information and defining the capacitors behaviour as accurate as possible, this work defines three different types of capacitors according to their connections represented inside the relay diagrams: Positive-negative, Negative-positive and Bipolar capacitors. Figure 3.8 presents an example of each of these capacitors.

The Positive-negative capacitor is the one whose left and right plates can only assume the

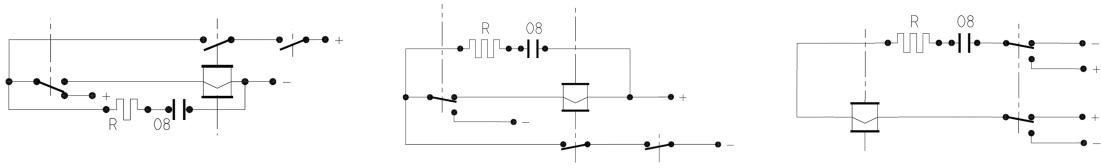


Figure 3.8 – An example of each capacitor type: Positive-negative, Negative-positive and Bipolar, respectively.

positive and negative charges, respectively. As the capacitors in the SNCF relay diagrams always are depicted in horizontally drawn wires, it is possible to assume the existence of a left and a right plate. In order to define a capacitor as a Positive-negative one, either the left plate can only be connected to positive energy sources or the right plate can only be connected to negative energy sources. As a consequence of this configuration, the left and right plates can only assume a positive and negative configuration, respectively.

Similarly, a Negative-positive capacitor is the one whose left and right plates can only assume the negative and positive charges, respectively. So, either the left plate can only be connected to negative energy sources or the right plate can only connect with positive energy sources.

Positive-negative and Negative-positive capacitors can be both mathematically modelled with the logic presented in this work. By giving more information about the context of these components it is possible to reduce the possibilities of charges that may be assumed, so the logic of this component can be reduced to its charging/discharging states.

Regarding Bipolar capacitors, their plates can be connected to the positive or negative energy sources. As a result, these plates can assume different charges at different moments. Thus, in order to determine a Bipolar capacitor state and its plates charges, we must know its previous states and connections, which can be impractical with the logic we use in this work as mentioned earlier. So, these capacitors are not modelled in this work.

In order to formalise the Positive-negative and Negative-positive capacitors, one must initially have means to differentiate them. This differentiation between the capacitors type is presented in the Structural Definition SD11.

Structural Definition SD11 (Capacitors Structural Differentiation): In this work only Positive-negative and Negative-positive capacitors are formalised. In order to differentiate them, a function from the set of capacitors to the set of capacitors types can be defined. So, given the constant universal set $CapPossTypes$, such that:

$$CapPossTypes = \{Pos_neg, Neg_pos\},$$

which contains the capacitors possible types Pos_neg (Positive-negative) and Neg_pos (Negative-positive), it is possible to differentiate the capacitors by the function $CapacitorsTypes_{\rightarrow}$, defined below:

$$CapacitorsTypes_{\rightarrow} \in (Capacitors \rightarrow CapPossTypes).$$

As a total function, it assures that every capacitor must have one and only one defined type. Later, in a future extension of this work, the capacitors types set can be extended in order to consider the existence of bipolar capacitors.

In order to formalise the behaviour of these components it is necessary to consider each plate as a different component. A capacitor is formed by the combination of two plates. This structural relation between the capacitors and their plates may be formalised as presented in the Structural Definition SD12

Structural Definition SD12 (Capacitors Structural Definition): A capacitor is formed by the combination of two plates. Each plate has a fixed position in the capacitor, which can be *right* or *left*. The relation between the capacitors and their belonging plates as well as the definition of the plates positions inside a capacitor can be modelled in two different functions:

- $PlatesCapRel_{\rightarrow}$, from the set of plates to the set of capacitors, which relates the plates with the capacitors to which they belong;
- $PlatesPosition_{\rightarrow}$, from the set of capacitors plates to the set containing their possible positions (*right* or *left*).

So, given the constant universal set R_L defined before, these functions can be mathematically represented as:

$$\begin{aligned} PlatesCapRel_{\rightarrow} &\in (CapPlates \rightarrow Capacitors) \wedge \\ PlatesPosition_{\rightarrow} &\in (CapPlates \rightarrow R_L). \end{aligned}$$

As total functions they define that every capacitor plate belongs to one and only one capacitor, besides, every plate has a position defined. However, this structural definition allows capacitors to have more than two plates, which is not a possible configuration. Thus, in order to guarantee the system well-definedness, one must establish that each capacitor must have only two plates, one on the left side and other on the right side. This property can be mathematically described as:

$$\begin{aligned} \forall cap.(cap \in Capacitors \Rightarrow \exists p1, p2.(p1 \in CapPlates \wedge p2 \in CapPlates \wedge \\ PlatesCapRel_{\rightarrow}^{-1}[\{cap\}] = \{p1, p2\} \wedge \\ PlatesPosition_{\rightarrow}(p1) = right \wedge \\ PlatesPosition_{\rightarrow}(p2) = left \wedge)). \end{aligned}$$

Based on these structural information, it is possible to define one last structural definition. This is related to the disposition of capacitors plates to assume specific charges. Due to system configuration that limits the energy source poles that the capacitors plates can be connected to, one may consider the existence of a potential of these components to assume determined charges. This potential may be deduced from the type of capacitor and the position of the capacitor plates, as it is presented in the Structural Definition SD13. It is important to mention that this definition does not apply to bipolar capacitors, whose plates may assume both charges.

Structural Definition SD13 (Capacitors Plates Potential Charge): Based on the type of the capacitor and the position of the capacitor plate, it is possible to deduce the potential of the plates to assume determined charges. Given the constant universal set $Charges$ containing the

possible plates charges positive (*pos*) and negative (*neg*), such that:

$$Charges = \{pos, neg\},$$

the relation between the capacitors plates and the charges they tend to assume may be represented by the function $PlateCharge_{\rightarrow}$, from the set of capacitors plates to the set $Charges$, such that:

$$\begin{aligned} PlateCharge_{\rightarrow} &\in (CapPlates \rightarrow Charges) \wedge \\ PlateCharge_{\rightarrow} &= \{(cp, ch) | cp \in CapPlates \wedge ch \in Charges \wedge \\ &((CapacitorTypes_{\rightarrow}(PlatesCapRel_{\rightarrow}(cp)) = Pos_neg \wedge \\ &PlatesPositions_{\rightarrow}(cp) = left) \Rightarrow ch = pos) \wedge \\ &((CapacitorTypes_{\rightarrow}(PlatesCapRel_{\rightarrow}(cp)) = Pos_neg \wedge \\ &PlatesPositions_{\rightarrow}(cp) = right) \Rightarrow ch = neg) \wedge \\ &((CapacitorTypes_{\rightarrow}(PlatesCapRel_{\rightarrow}(cp)) = neg_pos \wedge \\ &PlatesPositions_{\rightarrow}(cp) = left) \Rightarrow ch = neg) \wedge \\ &((CapacitorTypes_{\rightarrow}(PlatesCapRel_{\rightarrow}(cp)) = neg_pos \wedge \\ &PlatesPositions_{\rightarrow}(cp) = right) \Rightarrow ch = pos)\} \end{aligned}$$

As a total function, it defines that every capacitor plate has a potential to assume a specific charge. Nonetheless, in a future extension of this formalisation that considers bipolar capacitors, this function may be adapted in order to be a partial function, as the plates of bipolar capacitors do not have a specific potential charge as the other types. In the form it is defined, the capacitors plates have a tendency to assume:

- a positive charge if it is in the right position of a positive-negative capacitor,
- a negative charge if it is in the left position of a negative-positive capacitor,
- a positive charge if it is in the right position of a negative-positive capacitor or
- a negative charge if it is in the left position of a negative-positive capacitor.

Given all this structural context, one may be able to define the capacitors behaviour. For this purpose, it is possible to consider that capacitors may assume two states: deactivated or activated. The former represents the discharged or the charging moments, i.e., when the capacitor cannot provide energy. The latter represents the charged and discharging moments, i.e., when the capacitor can act as an energy source. Nonetheless, before defining the capacitors states, it is important to define the passing of time for these components. Capacitors are similar to both timed activation and timed deactivation blocks as they need some time to activate and another time to deactivate. The capacitor passing of time is presented in the Behavioural Definition BD20.

Behavioural Definition BD20 (Capacitor Passing of Time State Definition): For each capacitor, the time necessary for their activation and deactivation may be abstracted by two functions indicating if the required time has passed or not. So, given the constant universal set \mathbb{B} of

Boolean values, the passing of time for the activation and deactivation of each capacitor may be represented as two functions from the set of capacitors to the boolean set, such that:

$$\begin{aligned} CapActPassOfTime_{\rightarrow} &\in (Capacitors \rightarrow \mathbb{B}) \wedge \\ CapDeactPassOfTime_{\rightarrow} &\in (Capacitors \rightarrow \mathbb{B}), \end{aligned}$$

where $CapActPassOfTime_{\rightarrow}$ indicates the passing of time necessary for the capacitors activation and $CapDeactPassOfTime_{\rightarrow}$ indicates the passing of time necessary for the capacitors deactivation.

Similarly to blocks, the capacitors states are defined based on their electrification and activation. These components may be electrified and not activated as well as they can be not electrified and still activated. This behaviour is caused by the time that this component takes in order to store and lose energy.

When electrified, a capacitor only activates (charges) after a certain time. Once activated and no longer electrified, this component only deactivates after another certain time. The electrified/not electrified and activated/deactivated states may be described based on Boolean values, where *true* represents that the capacitor is electrified or activated and *false* represents that this component is not electrified or deactivated. When activated, the capacitor is charged and may provide energy to the system. Otherwise, when the capacitor is deactivated, it is not charged and cannot provide energy. The capacitors state representation may be mathematically formalised as presented in the Behavioural Definition BD21.

Behavioural Definition BD21 (Capacitors States Representation): The capacitors states may be represented by total functions from the set of capacitors to the set of Boolean values, where the values *true* and *false* indicate that the capacitor is activated/electrified or deactivated/not electrified, respectively. So, given the constant universal set \mathbb{B} defined before, the capacitors states may be represented by the functions $CapIsElectrified_{\rightarrow}$ and $CapIsActivated_{\rightarrow}$, indicating whether the capacitors are electrified and activated, respectively. These functions are mathematically represented as:

$$\begin{aligned} CapIsElectrified_{\rightarrow} &\in (Capacitors \rightarrow \mathbb{B}) \wedge \\ CapIsActivated_{\rightarrow} &\in (Capacitors \rightarrow \mathbb{B}) \end{aligned}$$

In order to be electrified, the capacitor plates must be connected to the positive and negative energy sources (each one connected to a different energy source pole). When electrified for a certain time, this component is activated. When activated, if this component is no longer electrified, it remains activated for a another certain time. The conditions for the capacitors electrification and activation are formalised in the Behavioural Definition BD22.

Behavioural Definition BD22 (Capacitors States Definition): Capacitors are electrified when each of their plates are connected to a different electrical energy source pole. However, the activation of these components only occurs after a certain time that they are electrified. Once a capacitor is activated and is no longer electrified, it remains activated for a certain time and then it deactivates. The electrification and activation conditions for a capacitor *cap* may be

mathematically described as:

$$\begin{aligned} \text{CapIsElectrified}_{\rightarrow}(cap) \Leftrightarrow \\ \exists p, n, pl1, pl2, PPa_{\rightarrow}, NPa_{\rightarrow}. (p \in \text{PosSources} \wedge n \in \text{NegSources} \wedge pl1 \in \text{CapPlates} \wedge \\ pl2 \in \text{CapPlates} \wedge pl1 \neq pl2 \wedge \text{PlatesCapRel}_{\rightarrow}(pl1) = cap \wedge \text{PlatesCapRel}_{\rightarrow}(pl2) = cap \wedge \\ PPa_{\rightarrow} \in \text{path}(pl1, p, RC_{\leftrightarrow}) \wedge NPa_{\rightarrow} \in \text{path}(pl2, n, RC_{\leftrightarrow})) \end{aligned}$$

$$\begin{aligned} \text{CapIsActivated}_{\rightarrow}(cap) \Leftrightarrow \\ ((\text{CapIsElectrified}_{\rightarrow}(cap) \wedge \text{CapActPassOfTime}_{\rightarrow}(cap)) \vee \\ (\neg \text{CapIsElectrified}_{\rightarrow}(cap) \wedge \neg \text{CapDeactPassOfTime}_{\rightarrow}(cap))) \end{aligned}$$

Once activated, the capacitors may produce energy to the system. This impacts the system state definition as the capacitors may act as another energy source. So, the components electrification condition may be adapted again in order to consider these components. This is the complete electrification condition as it considers all the possible energy sources presented in this work. Once again, this electrification condition is valid in order to state whether non-timed independent components are electrified or not, i.e., it defines the activation condition for monostable relays, bistable relays coils, and outputs. The Complete Electrification Condition is presented in the Behavioural Definition BD23.

Behavioural Definition BD23 (Complete Electrification Condition): Non-temporised independent components are electrified when connected to:

1. A positive and a negative energy sources;
2. A positive energy source and a negative capacitor plate of an activated capacitor;
3. A negative energy source and a positive capacitor plate of an activated capacitor;
4. The positive and the negative plates of activated capacitors;
5. An energy source and a block independent connection;
6. A capacitor plate of an activated capacitor, and a block independent connection;
7. Two block independent connections;
8. Two block dependent connections of an activated block.

Based on these conditions, it is possible to mathematically define that the conditions for a

component *comp* to be electrified as:

$electrified(comp) \Leftrightarrow$

$$\begin{aligned}
& ((\exists ps, ns, Pa_{\rightarrow}. (ps \in PosSources \wedge ns \in NegSources \wedge \\
& \quad Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& (\exists ps, ns, Pa_{\rightarrow}. (ps \in PosSources \wedge ns \in CapPlates \wedge PlateCharge_{\rightarrow}(ns) = neg \wedge \\
& \quad CapIsElectrified_{\rightarrow}(PlatesCapRel_{\rightarrow}(ns)) \wedge Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& (\exists ps, ns, Pa_{\rightarrow}. (ps \in CapPlates \wedge ns \in NegSources \wedge PlateCharge_{\rightarrow}(ps) = pos \wedge \\
& \quad CapIsElectrified_{\rightarrow}(PlatesCapRel_{\rightarrow}(ps)) \wedge Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& (\exists ps, ns, Pa_{\rightarrow}. (ps \in CapPlates \wedge ns \in CapPlates \wedge PlateCharge_{\rightarrow}(ps) = pos \wedge \\
& \quad CapIsElectrified_{\rightarrow}(PlatesCapRel_{\rightarrow}(ps)) \wedge PlateCharge_{\rightarrow}(ns) = neg \wedge \\
& \quad CapIsElectrified_{\rightarrow}(PlatesCapRel_{\rightarrow}(ns)) \wedge Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& (\exists bl, es, Pa_{\rightarrow}. (bl \in blocks \wedge es \in (PosSources \cup NegSources) \wedge \\
& \quad Pa_{\rightarrow} \in path(bl, es, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge (bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
& (\exists bl, es, Pa_{\rightarrow}. (bl \in blocks \wedge es \in CapPlates \wedge CapIsElectrified_{\rightarrow}(PlatesCapRel_{\rightarrow}(es)) \wedge \\
& \quad Pa_{\rightarrow} \in path(bl, es, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge (bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
& (\exists bl, Pa_{\rightarrow}. (bl \in blocks \wedge Pa_{\rightarrow} \in cycle(bl, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge \\
& \quad (bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow} \wedge (bl, last(front(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
& (\exists bl, Pa_{\rightarrow}. (bl \in blocks \wedge Pa_{\rightarrow} \in cycle(bl, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge BlockIsActivated_{\rightarrow}(bl) \wedge \\
& \quad (bl, first(tail(Pa_{\rightarrow}))) \in BlockDepConn_{\leftrightarrow} \wedge (bl, last(front(Pa_{\rightarrow}))) \in BlockDepConn_{\leftrightarrow})))
\end{aligned}$$

where the new conditions for the components electrification based on the use of capacitors are highlighted in yellow.

This Complete Electrification Condition is a general condition that defines whether any non-timed independent electrical component is electrified or not. However, in cases where capacitors and blocks are not used, this condition may be adapted or the other defined electrification conditions may be used.

Although resistors have no structural or behavioural detail that must be formalised, they may have a structural importance for the system. These components are used in order to control the time for the capacitors to charge, which has no importance in this formalisation as the time is abstracted as a system input. However, as part of the model, these components may still be modelled as a way to make the structural formalisation more accurate and close to reality.

Before finalising this formalisation, it is important to consider a variation of the energy sources that causes the lights to flash. This variation is analysed and presented in the next section.

3.3 Flashing Lights: An Energy Source Variation

The energy sources used to electrify signal lights in the SNCF systems have an alternating current. As the majority of the French relay-based RIS components use a direct current, the AC energy sources are only used for the electrification of some specific components, like some determined outputs. Although the logic of the system tends to be the same in both type of currents, some adaptations must be considered in order to deal with the energy source variation. Furthermore, the variety of the energy sources increases when the existence of flashing energy sources are considered. This last component type allows signal lights to flash, giving new meanings to them.

By considering the existence of AC energy sources, it is possible to extend the formalisation in order to describe three new types of components: AC energy sources, AC energy source returns and the flashing AC energy sources. So it is possible to define the sets *ACSources*, *ACReturns* and *FlashACSources* which are subsets of the *Components* sets and containing the AC energy sources, AC energy source returns and the flashing AC energy sources, respectively. This is mathematically defined as:

$$\begin{aligned} ACSources &\in Components \wedge \\ ACReturns &\in Components \wedge \\ FlashACSources &\in Components. \end{aligned}$$

The logic for the activation of components connected to these energy sources is the same presented in the Behavioural Definitions BD7, BD19 and BD23. However, it is necessary to extend them in order to consider that the components are also activated when they are in the paths between:

- AC energy sources and AC energy source returns, or
- Flashing AC energy sources and AC energy source returns.

For the system behavioural analysis, it may also be important to verify if the output components are connected to a fixed or a flashing energy source in order to determine if the signal lights are flashing or not. This is because fixed and flashing signals have different meanings that can affect the system safety. In order to analyse the type of AC current flowing inside an output, one may verify if there is a path between this component and a flashing AC energy source. In a positive scenario, the component has a flashing AC flowing through it.

Furthermore, the existence of AC current may cause some adaptations on the block format and activation conditions. When both direct and alternating electrical current are used in the system, the block may have direct connections to the direct (24V) and alternating (400Hz) current energy sources, as presented in Figure 3.9. In this case, the block activation may also result from its connection with an AC energy source. Once activated, the block is able to generate a direct current in order to activate relays, for instance. In this context, the block may be activated when connected to any type of energy source though an independent connection.

The use of alternating current energy sources is a variation of the relay-based RIS that requires an extension of the formalisation presented in this work. In this case, it is possible to extend the components electrification conditions and the blocks electrification conditions

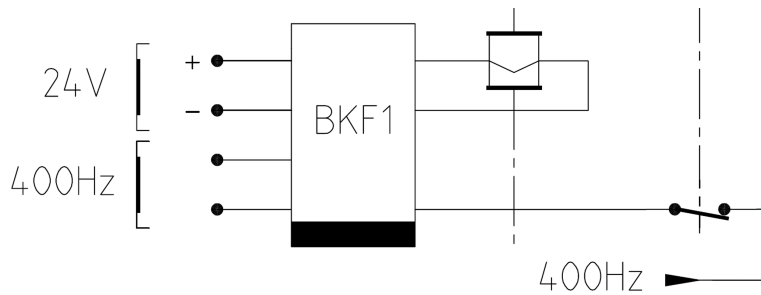


Figure 3.9 – Block adapted to the use of both direct and alternating currents.

presented in the Behavioural Definitions BD23 and BD18 in order to consider the existence of the new energy sources. These new conditions are described in the Behavioural Definitions BD24 and BD25.

Behavioural Definition BD24 (Extended Electrification Condition): In a system that allows the existence of AC and DC energy sources, non-temporised independent components are electrified when connected to:

1. A positive and a negative energy sources;
2. A positive energy source and a negative capacitor plate of an activated capacitor;
3. A negative energy source and a positive capacitor plate of an activated capacitor;
4. The positive and the negative plates of one or more activated capacitors;
5. An AC energy source and an AC energy source return;
6. A flashing AC energy source and an AC energy source return;
7. An AC or DC energy source and a block independent connection;
8. A capacitor plate of an activated capacitor, and a block independent connection;
9. Two block independent connections;
10. Two block dependent connections of an activated block.

Based on these conditions, it is possible to mathematically define the conditions for a component

comp to be electrified as:

$electrified(comp) \Leftrightarrow$

$$\begin{aligned}
& ((\exists ps, ns, Pa_{\rightarrow}. (ps \in PosSources \wedge ns \in NegSources \wedge \\
& \quad Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& (\exists ps, ns, Pa_{\rightarrow}. (ps \in PosSources \wedge ns \in CapPlates \wedge PlateCharge_{\rightarrow}(ns) = neg \wedge \\
& \quad CapIsElectrified_{\rightarrow}(PlatesCapRel_{\rightarrow}(ns) \wedge Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& (\exists ps, ns, Pa_{\rightarrow}. (ps \in CapPlates \wedge ns \in NegSources \wedge PlateCharge_{\rightarrow}(ps) = pos \wedge \\
& \quad CapIsElectrified_{\rightarrow}(PlatesCapRel_{\rightarrow}(ps) \wedge Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& (\exists ps, ns, Pa_{\rightarrow}. (ps \in CapPlates \wedge ns \in CapPlates \wedge PlateCharge_{\rightarrow}(ps) = pos \wedge \\
& \quad CapIsElectrified_{\rightarrow}(PlatesCapRel_{\rightarrow}(ps) \wedge PlateCharge_{\rightarrow}(ns) = neg \wedge \\
& \quad CapIsElectrified_{\rightarrow}(PlatesCapRel_{\rightarrow}(ns) \wedge Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& ((\exists ps, ns, Pa_{\rightarrow}. (ps \in ACSources \wedge ns \in ACReturns \wedge \\
& \quad Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& ((\exists ps, ns, Pa_{\rightarrow}. (ps \in FlashACSources \wedge ns \in ACReturns \wedge \\
& \quad Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
& (\exists bl, es, Pa_{\rightarrow}. (bl \in blocks \wedge \\
& \quad es \in (PosSources \cup NegSources \cup ACSources \cup ACReturns \cup FlashACSources) \wedge \\
& \quad Pa_{\rightarrow} \in path(bl, es, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge (bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
& (\exists bl, es, Pa_{\rightarrow}. (bl \in blocks \wedge es \in CapPlates \wedge CapIsElectrified_{\rightarrow}(PlatesCapRel_{\rightarrow}(es) \wedge \\
& \quad Pa_{\rightarrow} \in path(bl, es, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge (bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
& (\exists bl, Pa_{\rightarrow}. (bl \in blocks \wedge Pa_{\rightarrow} \in cycle(bl, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge \\
& \quad (bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow} \wedge (bl, last(front(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
& (\exists bl, Pa_{\rightarrow}. (bl \in blocks \wedge Pa_{\rightarrow} \in cycle(bl, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge BlockIsActivated_{\rightarrow}(bl) \wedge \\
& \quad (bl, first(tail(Pa_{\rightarrow}))) \in BlockDepConn_{\leftrightarrow} \wedge (bl, last(front(Pa_{\rightarrow}))) \in BlockDepConn_{\leftrightarrow}))).
\end{aligned}$$

where the new conditions for the components electrification based on the use of AC energy sources are highlighted in yellow.

Behavioural Definition BD25 (Extended Blocks Electrification Condition): In a system that allows the existence of AC and DC energy sources, a block is electrified if there is a cycle that begins and finishes in the block passing through its independent connections or if there is a path between the block and an energy source (positive, negative, AC sources or AC source return) that passes through an independent connection. These conditions may be generally mathematically

defined for all the system blocks as:

$$\begin{aligned}
& \text{BlockIsElectrified}_{\rightarrow}(bl) \Leftrightarrow \\
& \exists es, Pa_{\rightarrow}. (es \in (\text{PosSources} \cup \text{NegSources} \cup \text{ACSources} \cup \text{ACReturns})) \wedge \\
& \quad (((Pa_{\rightarrow} \in \text{path}(bl, es, RC_{\leftrightarrow}) \wedge (bl, \text{first}(\text{tail}(Pa_{\rightarrow}))) \in \text{BlockIndConn}_{\leftrightarrow})) \vee \\
& \quad (Pa_{\rightarrow} \in \text{cycle}(bl, RC_{\leftrightarrow}) \wedge ((bl, \text{first}(\text{tail}(Pa_{\rightarrow}))) \in \text{BlockIndConn}_{\leftrightarrow}) \wedge \\
& \quad \quad ((bl, \text{last}(\text{front}(Pa_{\rightarrow}))) \in \text{BlockIndConn}_{\leftrightarrow})))
\end{aligned}$$

A last behavioural definition is concerned with the fact that an output (a light signal, more specifically) may be flashing or not. As the flashing light is a possible behaviour of the system, it is important to formalise it. Furthermore, this formalisation may be useful for the verification of the system behaviour in order to guarantee its safety, as flashing lights have a different semantic meaning than the fixed ones. The conditions for an output to be flashing is presented in the Behavioural Definition BD26.

Behavioural Definition BD26 (Outputs Flashing Condition): An output is flashing if it is electrified and connected to a flashing AC energy source while it is not connected to a normal AC energy source. This may be mathematically formalised for all outputs as:

$$\begin{aligned}
& \text{isFlashing}(out) \Leftrightarrow \\
& \quad (\text{electrified}(out) \wedge \exists es, Pa_{\rightarrow}. (es \in (\text{FlashACSource}) \wedge Pa_{\rightarrow} \in \text{path}(out, es, RC_{\leftrightarrow}))) \wedge \\
& \quad \neg \exists es, Pa_{\rightarrow}. (es \in (\text{ACSource}) \wedge Pa_{\rightarrow} \in \text{path}(out, es, RC_{\leftrightarrow}))
\end{aligned}$$

Based on the formalisation of the relay-based RIS presented in this work, it is possible to formalise a great variety of systems. This formalisation may also be used in order to perform some structural well-definedness and safety behaviour verification. The system verification using this formalisation is discussed in the next section.

3.4 Formalisation Support for the System Verification

The structural and the behavioural relay-based RIS formalisation presented in this chapter can be used in order to verify the relay diagrams structure well-definedness and the safety of the system behaviour, respectively. This can be made using the logical descriptions as basis for the definition of well-definedness and safety conditions that can be formally verified. This section presents how the relay-based RIS formalisation presented in this chapter may be used in order to perform verifications about the system well-definedness and safety.

3.4.1 Structural Well-definedness Verification

As the structure of the relay-based RIS is able to be mathematically defined, it is also able to be formally analysed in order to guarantee its correctness. The system behaviour cannot be properly analysed if the structure can be faulty, since a faulty structure generates a non-viable system. Although some well-definedness conditions can be generalised to every relay-based system, each company has different patterns for modelling relay diagrams, so the structural

well-definedness conditions presented in this section may be adapted as a way to support a structural verification in other different approaches. Indeed, the capabilities of the mathematical expressions are almost limitless, so the conditions presented here can be extended and adapted in order to create new conditions.

In fact, the structural verification of these systems can be more cost-effective if the generation of the mathematical expressions and the structural verification are made by tools. This is because the system manual mathematical specification may be a hard and time consuming process. The creation of a tool for the automatic generation and verification of the system structural and behavioural mathematical definition is a future work of this thesis. The objective is to use the structural verification presented in this section as a support in order to guarantee that the relay-diagrams are well defined before transforming them into a formal specification for behavioural verification purposes. Thus, the well-definedness condition presented in this section aims to support the verification of the relay diagrams structural correctness.

In this work we can divide the well-definedness conditions into two different types: general and specific conditions. The former define properties that must be met by every system independently of the companies modelling patterns. These general well-definedness conditions are related, for instance, to the fact that every bistable relay has only two coils, the blocks dependent and independent connections are different or that capacitors have always two plates. All these conditions have been presented in this chapter as part of the structural definitions. Nonetheless, they are also considered as well-definedness conditions since they were created with the objective of guaranteeing the structural correctness of the model.

Another example of general well-definedness condition presented before is related to the relay diagrams basic structure. In this context, it is important to guarantee that the components subsets are mutually disjoint and that their union forms the *Components* set. This condition is expressed as:

$$Components = \bigcup ComponentsSubsets \wedge$$

$$\forall S1, S2. ((S1 \in ComponentsSubsets \wedge \\ S2 \in ComponentsSubsets \wedge \\ S1 \neq S2) \Rightarrow S1 \cap S2 = \emptyset).$$

This expression states a condition that must be met in order to assure that components have not a double function.

The specific well-definedness conditions specify the properties that the system must meet according to a company design patterns. These properties are not generalised as they are related to the relay diagram structure. One simple example is the definition of a condition that states that the set of wires and components must be disjoint. This is an important condition in our relay-based RIS model in order to guarantee that the name given to the components are not the same as the names given to the wires during the formalisation. This condition may be defined as:

$$Wires \cap Components = \emptyset.$$

Regarding the basic structure of the relay diagram in the form of a graph, it is possible to define some specific structural properties in order to guarantee its well-definedness. For instance, it is possible to determine the quantity of connections of each component as a way to guarantee that the components connections are well defined in the incidence function of the graph. The system behavioural definitions depend on the fact that each component has a certain number of connections and the graph representing the relay diagram must meet these constraints. The Table 3.2 presents the number of connections for each component in the relay diagrams used by SNCF, which may be different in other companies design patterns. This table also presents the logical conditions that may be used in the system formalisation as a way to enforce the quantity of connections for each component. These expressions are based on the graph notion of vertex degree presented in Section 1.3.4.

Table 3.2 – Number of allowed connections for each type of component.

Component	Connections	Logical definition
Positive sources	1	$\forall x.(x \in PosSources \Rightarrow (degree(x, Incidence_{\rightarrow}) = 1))$
Negative sources	1	$\forall x.(x \in NegSources \Rightarrow (degree(x, Incidence_{\rightarrow}) = 1))$
Capacitors plates	1	$\forall x.(x \in CapPlates \Rightarrow (degree(x, Incidence_{\rightarrow}) = 1))$
Buttons	2	$\forall x.(x \in Buttons \Rightarrow (degree(x, Incidence_{\rightarrow}) = 2))$
Lever contacts	2	$\forall x.(x \in LeverContacts \Rightarrow (degree(x, Incidence_{\rightarrow}) = 2))$
Monostable relays	2	$\forall x.(x \in MonostableRelays \Rightarrow (degree(x, Incidence_{\rightarrow}) = 2))$
Bistable relay coils	2	$\forall x.(x \in BistableCoils \Rightarrow (degree(x, Incidence_{\rightarrow}) = 2))$
outputs	2	$\forall x.(x \in Outputs \Rightarrow (degree(x, Incidence_{\rightarrow}) = 2))$
Monostable contacts	2 - 3	$\forall x.(x \in MonostableContacts \Rightarrow (degree(x, Incidence_{\rightarrow}) \in 2..3))$
Bistable contacts	2 - 3	$\forall x.(x \in BistableContacts \Rightarrow (degree(x, Incidence_{\rightarrow}) \in 2..3))$
Junctions	2 - 4	$\forall x.(x \in Junctions \Rightarrow (degree(x, Incidence_{\rightarrow}) \in 2..4))$
blocks	5 - 8	$\forall x.(x \in Blocks \Rightarrow (degree(x, Incidence_{\rightarrow}) \in 5..8))$
AC energy sources	1	$\forall x.(x \in ACSources \Rightarrow (degree(x, Incidence_{\rightarrow}) = 1))$
AC energy source returns	1	$\forall x.(x \in ACReturns \Rightarrow (degree(x, Incidence_{\rightarrow}) = 1))$
Flashing AC energy source	1	$\forall x.(x \in FlashACSources \Rightarrow (degree(x, Incidence_{\rightarrow}) = 1))$

Another structural well-definedness property specific to out context may guarantee that the incidence function does not contain any component that is not supposed to be used. This is because some components are composed by sub-components. In this context, only the sub-components are represented in the incidence function. So, it is possible to define the logical expression:

$$\forall comp.(comp \in Components \wedge (comp \in BistableRelays \cup Levers \cup Capacitors) \Rightarrow (comp \notin (dom(ran(Incidence_{\rightarrow})) \cup ran(ran(Incidence_{\rightarrow})))));$$

making it clear that bistable relays, levers and capacitors are not connected in the graph incidence function.

Regarding the inputs of the system, some structural well-definedness properties may also be defined. Due to the complexity of the levers structures it is important to establish some conditions for their structural definitions. One of these conditions is that every lever contact must have a configuration defined for each lever configuration. Besides, the configuration of a lever contact must never be the same in both of its related lever configurations. These conditions may be guaranteed by defining that:

$$ConfigRel_{\leftrightarrow}^{-1} \in ((LeverContacts \times \mathbb{B}) \rightarrow (Levers \times LeverConfig)),$$

which, by using a total function, states that every lever contact configuration is always related to one, and only one, lever configuration.

Many other structural well-definedness properties may be defined in order to assure that the modelled diagram and its formalised version are conform to to the structural expectations. These definitions are not exhaustive and they can always be extended in order to support other contexts and the definition of new properties.

While the logical expressions for the verification of the system structure can be defined based on the knowledge about the electrical circuits models, the system safety behaviour verification requires a background about the system behaviour, environment and context.

3.4.2 Behavioural Safety Conditions Definition

The behavioural relay-diagram description based on mathematical expressions allows one to prove the system safety by defining safety conditions which may be logically analysed. These conditions are related to the existence or non-existence of specific states. For instance, in the ITCS case study, one must guarantee that two components are never electrified at the same time, so they cannot be activated at the same time. So, one may define logical expressions in order to relate the state of components as a way to guarantee that safety conditions are met at any system state.

In this context, it is important to note that each relay diagram models a specific system in a particular context. Thus, one cannot create general safety conditions than can be applied to every case study. Generally, these safety conditions are created by experts based on the analysis of the system context and the relay diagram. As a consequence, each safety condition is specific to a particular case study.

In order to define these safety conditions, any logical expression that relates two components may be used. Some examples of expressions are the guarantee that:

1. Two distinct components x and y are never electrified at the same time:

$$\neg(electrified(x) \wedge electrified(y));$$

2. Two components x and y are always electrified and not electrified at the same time:

$$electrified(x) \Leftrightarrow electrified(y);$$

3. If one component x electrifies, another component y must also be electrified:

$$electrified(x) \Rightarrow electrified(y).$$

These are some examples of simple logical expressions that may be used in order to assure that, respectively:

1. two signals are never green at the same time allowing trains to enter in the same tracks, which may avoid a frontal collision;
2. Every time that a turnout is turned, a light signal near it must enforce the decreasing of the train speed, which may avoid a derailment;
3. If a pedal detects the presence of a train in a track, the light signals close to this track must enforce the velocity decrease or the stop of other trains, which may avoid rear collisions.

Based on the system logical structural and behavioural formalisation and the state of each component, it is possible to determine many other safety conditions in order to guarantee the system safety. However, knowledge about the relation between the system and the environment is necessary as a way to determine how the relation between components may affect the safety. This expertise dependency for the safety conditions definition cannot be avoided at this level, since it is not possible to predict the impact of each component state in the real field as the environment is not formalised. A solution based on the creation of a more abstracted model focused on the simulation of the system environment is discussed as a future work in the conclusion of this thesis. By relating the physical components states with an abstract model focused on the trains position and velocity, it may be possible to detect which components configurations may cause accidents.

The creation of safety conditions based on the behavioural logic presented in this chapter may be manually verified based on the logical analysis. If there is the possibility that a safety condition is not met considering the possible system states, this system may not be considered safe. In this context, it is important to notice that the verification in this case is compositional at the diagram level as it is able to verify the safety of each diagram according to every possible input value. This same verification can be repeated when composing two or more diagrams. Furthermore, as specification and verification supporting tools may be used, these safety conditions may have other uses. One may use, for instance, a SAT-Solver, which analyses if there is a set of values for each system variable that satisfy every condition. In this case, a negation of the safety condition may be used as part of the system model, so the SAT-Solver may show if there is a set of values that satisfy this condition. Thus, if the answer is positive, the system cannot be considered safe.

By using a formal method focused on the specification of concurrent systems like CSP, for instance, one may define assertions in order to guarantee that a determined state defined by the safety condition is not reached, which can be verified by a model checker (more details about it is presented in the future works section in the end of this thesis). A similar approach can be seen in the B-method, where the safety conditions are defined as part of the specification invariant. Then a model checker can be used in order to guarantee that no state will be inconsistent with the invariant. Then, a logical verification based on the B-method proof obligations can be also used in order to guarantee that the state evolution cannot reach an inconsistent state.

As a way to exemplify the structural and behavioural formalisation presented in this chapter, the next section presents the formalisation of the ITCS case study. Furthermore, the manual verification of this system is presented, which states how the logic presented can be used as a way to make a manual formal verification of the system safety.

3.5 Case Study Specification and Analysis

Based on the formalisation of the relay-based RIS structure and behaviour presented in this chapter, it is possible to formalise the ITCS industrial example. In order to mathematically describe this system, one may specialise its structure according to the definitions presented in Sections 3.2 and 3.2.3. Then, together with the general behavioural formalisation description, the system is formalised and it can be verified by logically analysing it regarding safety conditions. This section is focused on presenting how the formalisation detailed in this chapter can be used in order to formalise and verify the ITCS case study. Before any behavioural description, it is important to logically define the system structure.

3.5.1 Structural Formalisation

Regarding the structural formalisation of the system, this chapter presents a mathematical model that describes the general physical relations between components. This model may then be specialised for each different system by adding the information about the components, their types and their connections as described inside the relay diagram. Thus it is possible to manually translate the relay diagram schema into the mathematical structural model.

The general structural model consists of all the structural definitions mathematical expressions presented in this chapter. So, considering the existence of the sets *Components* and *Wires*, which are the base of the structural formalisation, the general relay-based RIS structural model is defined based on the sets, relations and functions presented in the structural definitions. In this context, the graph incidence function and the components subsets are defined as:

$$\begin{aligned}
 Incidence_{\rightarrow} &\in (Wires \rightarrow (Components \times Components)) \quad \wedge \\
 PosSources &\subseteq Components \quad \wedge \quad NegSources \subseteq Components \quad \wedge \\
 Buttons &\subseteq Components \quad \wedge \quad Levers \subseteq Components \quad \wedge \\
 LeverContacts &\subseteq Components \quad \wedge \quad MonostableContacts \subseteq Components \quad \wedge \\
 BistableContacts &\subseteq Components \quad \wedge \quad Junctions \subseteq Components \quad \wedge \\
 Capacitors &\subseteq Components \quad \wedge \quad CapPlates \subseteq Components \quad \wedge \\
 resistors &\subseteq Components \quad \wedge \quad Blocks \subseteq Components \quad \wedge \\
 Outputs &\subseteq Components \quad \wedge \quad MonostableRelays \subseteq Components \quad \wedge \\
 BistableRelays &\subseteq Components \quad \wedge \quad BistableCoils \subseteq Components.
 \end{aligned}$$

Based on these basic structures, it is possible to define the structural relation between the components, their types and their states as:

$$\text{LeverConfig} = \{\text{config}_a, \text{config}_b\} \wedge \mathbb{B} = \{\text{true}, \text{false}\} \wedge$$

$$\text{LeverContactsRel}_{\rightarrow} \in (\text{LeverContacts} \rightarrow \text{Levers}) \wedge$$

$$\text{ConfigRel}_{\leftrightarrow} \in ((\text{Levers} \times \text{LeverConfig}) \leftrightarrow (\text{LeverContacts} \times \mathbb{B})) \wedge$$

$$\text{R_L} = \{\text{right}, \text{left}\} \wedge \text{U_D} = \{\text{up}, \text{down}\} \wedge$$

$$\text{MonoContactsConn}_{\rightarrow} \in ((\text{MonostableContacts} \times \text{U_D}) \rightarrow \text{Wires}) \wedge$$

$$\text{BistContactsConn}_{\rightarrow} \in ((\text{BistableContacts} \times \text{R_L}) \rightarrow \text{Wires}) \wedge$$

$$\text{BistableCoilsRel}_{\rightarrow} \in (\text{BistableCoils} \rightarrow \text{BistableRelays}) \wedge$$

$$\text{CoilSide}_{\rightarrow} \in (\text{BistableCoils} \rightarrow \text{R_L}) \wedge$$

$$\text{MonoRelayContactsRel}_{\rightarrow} \in (\text{MonostableContacts} \rightarrow \text{MonostableRelays}) \wedge$$

$$\text{BistRelayContactsRel}_{\rightarrow} \in (\text{BistableContacts} \rightarrow \text{BistableRelays}) \wedge$$

$$\text{BlockPossTypes} = \{\text{timed_act}, \text{timed_deact}\} \wedge \text{BlockTypes}_{\rightarrow} \in (\text{Blocks} \rightarrow \text{BlockPossTypes}) \wedge$$

$$\text{BlockDepConn}_{\leftrightarrow} \in (\text{Blocks} \leftrightarrow \text{Components}) \wedge \text{BlockIndConn}_{\leftrightarrow} \in (\text{Blocks} \leftrightarrow \text{Components}).$$

By specifying the general model, it is possible to define the structural formalisation of the ITCS example. In this case, the relay diagram must be interpreted and the physical connections between the components and their graphical differentiation must be formalised according to the general model expressions.

Initially, it is important to define the *Components* and the *Wires* sets. These sets are the basis of the entire formalisation and they contain all the sets and wires presented in the diagram. However, as generally the wires and some components are not named, it is important to name them so the system structure can be formalised. In this work, the names of the ITCS components and wires are defined according to the following logic:

- The components whose names are depicted in the relay diagram have their name maintained, but the spaces between the words and letters are replaced by underscores ("_");
- Contacts are named after their relays followed by "_x", where x is a natural number given to each contact in an ascending order according to its top-down appearance inside the relay diagram;
- Bistable relays coils are named after their relays followed by "_x", where x represents the position of the coil (*right* or *left*);
- Similarly to the contacts, lever contacts are named after their related levers followed by "_x", where x is a natural number given to each contact in an ascending order according to

its top-down appearance inside the relay diagram;

- As all buttons in the ITCS example are related to levers, these buttons are named after these levers followed by "*button*";
- Positive energy sources, negative energy sources, junctions and wires are named in the formats "*Px*", "*Nx*", "*Jx*" and "*Wx*", respectively, where *x* is a natural number.

Following this logic, the name of each component and wire of the ITCS example is defined as presented in Figure 3.10, where the wires names are presented in red and the components names are presented in green.

An important information that must be considered in the model instantiation is the universal sets, which contains the most basic information that are the basic building blocks of the formal model. As well as everything in the structural model, these sets are constant, and their information must never change during the system execution. In the ITCS example, the universal sets are the *LeverConfig*, \mathbb{B} , *U_D*, *R_L* and *BlockPossTypes* sets defined in the general model, together with the specific *Components* and *Wires* sets. These are all the basic sets that can be explicitly defined. As presented in the structural and behavioural models, the universal sets are:

$$\begin{aligned} \text{LeverConfig} &= \{\text{config}_a, \text{config}_b\} \wedge \mathbb{B} = \{\text{true}, \text{false}\} \wedge U_D = \{\text{up}, \text{down}\} \wedge \\ R_L &= \{\text{right}, \text{left}\} \wedge \text{BlockPossTypes} = \{\text{timed_act}, \text{timed_deact}\} \end{aligned}$$

In the ITCS case study, the components and wires sets can be explicitly defined as:

$$\begin{aligned} \text{Components} &= \{\text{KIT_C_911}, \text{KIT_C_CSS}, \text{SS_E_V2}, \text{INT_AC_V2}, \text{PG_911}, \text{EIT_C_CSS}, \\ &\text{C_CSS_V2}, \text{EIT_C_CSS_right}, \text{EIT_C_CSS_left}, \text{C_CSS_V2_right}, \text{C_CSS_V2_left} \\ &\text{P1}, \text{P2}, \text{P3}, \text{P4}, \text{P5}, \text{P6}, \text{P7}, \text{P8}, \text{P9}, \text{P10}, \text{P11}, \text{N1}, \text{N2}, \text{N3}, \text{N4}, \text{N5}, \text{N6}, \text{N7}, \text{L_C_CSS}, \\ &\text{L_C_CSS_button}, \text{L_C_CSS_1}, \text{L_C_CSS_2}, \text{L_C_CSS_3}, \text{L_ITCS}, \text{L_ITCS_button}, \\ &\text{L_ITCS_1}, \text{L_ITCS_2}, \text{J1}, \text{J2}, \text{J3}, \text{J4}, \text{J5}, \text{J6}, \text{J7}, \text{J8}, \text{J9}, \text{J10}, \text{TA_SS_E_V2}, \text{KIT_C_CSS_1}, \\ &\text{SS_E_V2_1}, \text{SS_E_V2_2}, \text{SS_E_V2_3}, \text{PG_911_1}, \text{PG_911_2}, \text{PG_911_3}, \text{EIT_C_CSS_1}, \\ &\text{EIT_C_CSS_2}, \text{EIT_C_CSS_3}, \text{EIT_C_CSS_4}, \text{C_CSS_V2_1}, \text{C_CSS_V2_2}, \text{C_CSS_V2_3}, \\ &\text{C_CSS_V2_4}, \text{INT_AC_V2}, \text{INT_AC_V2_1}, \text{INT_AC_V2_2}, \text{INT_AC_V2_3}, \\ &\text{INT_AC_V2_4}, \text{INT_AC_V2_5}, \text{KAU_V2}, \text{KAU_V2_1}, \text{EIT_C_912}, \text{EIT_C_912_1}, \\ &\text{EPA_C_CSS}, \text{EPA_C_CSS_1}, \text{EPA_C_CSS_2}, \text{EPA_C_911}, \text{EPA_C_911_1}, \text{EPA_C_912}, \\ &\text{EPA_C_912_1}, \text{KAG_a_G}, \text{KAG_a_G_1}, \text{RPD_FA_C911}, \text{RPD_FA_C911_1}\} \\ \text{Wires} &= \{\text{W1}, \text{W2}, \text{W3}, \text{W4}, \text{W5}, \text{W6}, \text{W7}, \text{W8}, \text{W9}, \text{W10}, \text{W11}, \text{W12}, \text{W13}, \text{W14}, \text{W15}, \text{W16}, \text{W17}, \\ &\text{W18}, \text{W19}, \text{W19}, \text{W20}, \text{W21}, \text{W22}, \text{W23}, \text{W24}, \text{W25}, \text{W26}, \text{W27}, \text{W28}, \text{W29}, \text{W30}, \text{W31}, \text{W32}, \\ &\text{W33}, \text{W34}, \text{W35}, \text{W36}, \text{W37}, \text{W38}, \text{W39}, \text{W40}, \text{W41}, \text{W42}, \text{W43}, \text{W44}, \text{W45}, \text{W46}, \text{W47}, \text{W48}, \\ &\text{W49}, \text{W50}, \text{W51}, \text{W52}, \text{W53}, \text{W54}, \text{W55}, \text{W56}, \text{W57}, \text{W58}, \text{W59}, \text{W60}, \text{W61}, \text{W62}, \text{W63}, \text{W64}, \\ &\text{W65}, \text{W66}, \text{W67}, \text{W68}, \text{W69}, \text{W70}, \text{W71}, \text{W72}\} \end{aligned}$$

The *Components* and the *Wires* sets are part of the graph definition that describes the

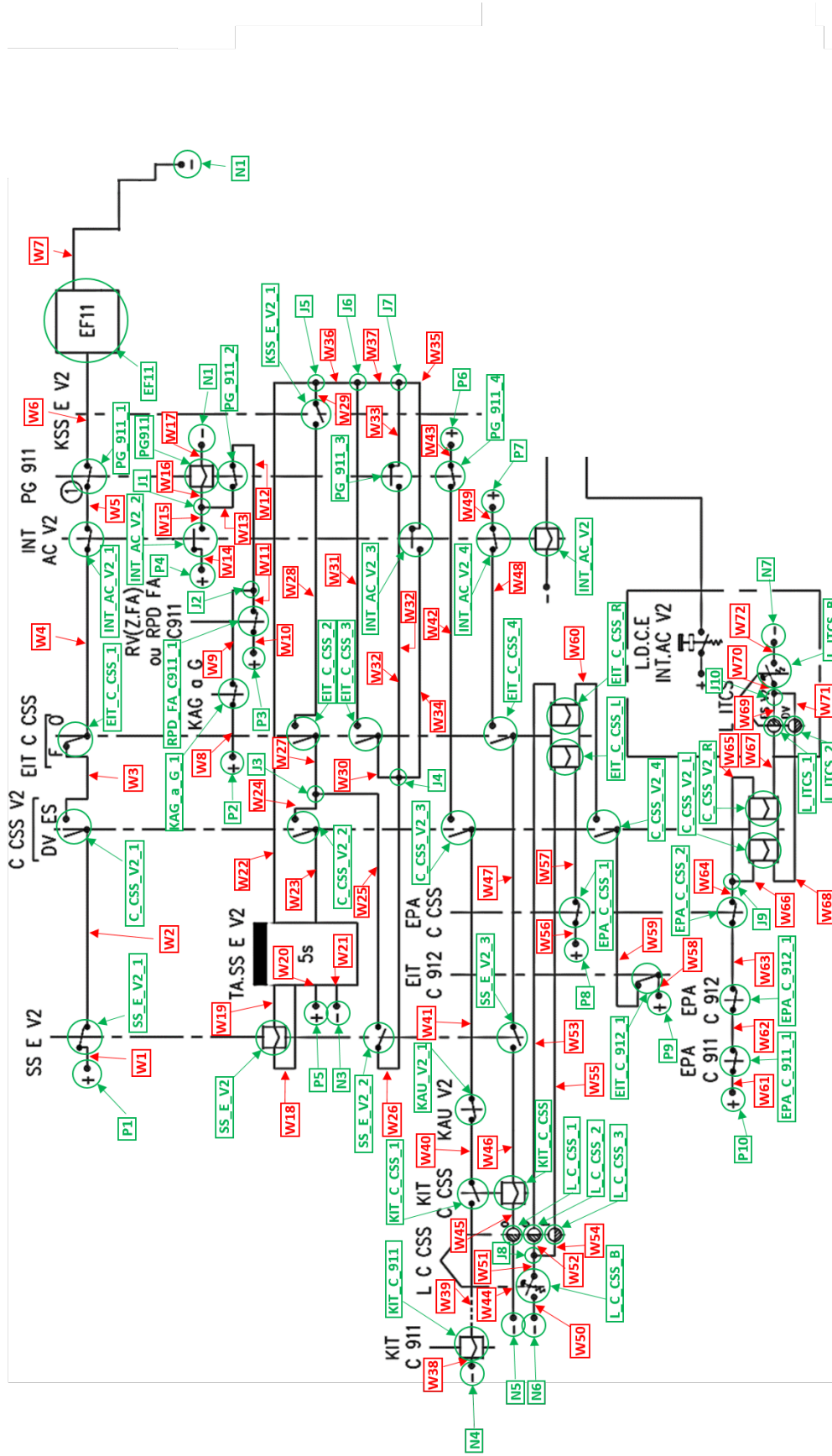


Figure 3.10 – ITCS example where components and wires are named.

relation between the electrical components as it is depicted in the relay diagram. These sets are defined according to the components and wires presented in the relay diagram and they are the basis for the definition of the graph incidence function. In order to specialise the ITCS incidence function, one must list the relations between each wire with couples of components, presenting the connection between the components mediated by wires. This function may then be specialised as the following:

$$\begin{aligned}
 Incidence_{\rightarrow} = & \{(W1,(P1,SS_E_V2_1)), (W2,(SS_E_V2_1,C_CSS_V2_1)), \\
 & (W3,(C_CSS_V2_1,EIT_C_CSS_1)), (W4,(EIT_C_CSS_1,INT_AC_V2_1)), \\
 & (W5,(INT_AC_V2_1,PG_911_1)), (W6,(PG_911_1,EF11)), (W7,(EF11,(N1))), \\
 & (W8,(P2,KAG_a_G)), (W9,(KAG_a_G,J2)), (W10,(P3,RPD_FA_C911_1)), \\
 & (W11,(RPD_FA_C911_1,J2)), (W12,(J2,PG_911_2)), (W13,(PG_911_2,J1)), \\
 & (W14,(P4,INT_AC_V2_2)), (W15,(INT_AC_V2_2,J1)), (W16,(J1,PG_911)), \\
 & (W17,(PG_911,N1)), (W18,(TA_SS_E_V2,SS_E_V2)), (W19,(SS_E_V2,TA_SS_E_V2)), \\
 & (W20,(P5,TA_SS_E_V2)), (W21,(N3,TA_SS_E_V2)), (W22,(TA_SS_E_V2,J5)), \\
 & (W23,(TA_SS_E_V2,C_CSS_V2_2)), (W24,(C_CSS_V2_2,J3)), \\
 & (W25,(J3,SS_E_V2_2)), (W26,(SS_E_V2_2,J4)), (W27,(J3,EIT_C_CSS_2)), \\
 & (W28,(EIT_C_CSS_2,KSS_E_V2_1)), (W29,(KSS_E_V2_1,J5)), \\
 & (W30,(J4,EIT_C_CSS_3)), (W31,(EIT_C_CSS_3,J6)), (W32,(J4,PG_911_3)), \\
 & (W33,(PG_911_3,J7)), (W34,(J4,INT_AC_V2_3)), (W35,(INT_AC_V2_3,J7)), \\
 & (W36,(J5,J6)), (W37,(J6,J7)), (W38,(N4,KIT_C_911)), \\
 & (W39,(KIT_C_911,KIT_C_CSS_1)), (W40,(KIT_C_CSS_1,KAU_V2_1)), \\
 & (W41,(KAU_V2_1,C_CSS_V2_3)), (W42,(C_CSS_V2_3,PG_911_4)), \\
 & (W43,(PG_911_4,P7)), (W44,(N5,L_C_CSS_1)), (W45,(L_C_CSS_1,KIT_C_CSS)), \\
 & (W46,(KIT_C_CSS,SS_E_V2_3)), (W47,(SS_E_V2_3,EIT_C_CSS_4)), \\
 & (W48,(EIT_C_CSS_4,INT_AC_V2_4)), (W49,(INT_AC_V2_4,P7)), \\
 & (W50,(N6,L_C_CSS_B)), (W51,(L_C_CSS_B,J8)), (W52,(J8,L_C_CSS_2)), \\
 & (W53,(L_C_CSS_2,EIT_C_CSS_R)), (W54,(J8,(L_C_CSS_3))), \\
 & (W55,(L_C_CSS_3,EIT_C_CSS_L)), (W56,(P8,EPA_C_CSS_1)), \\
 & (W57,(EPA_C_CSS_1,EIT_C_CSS_L)), (W58,(P9,EIT_C_912_1)), \\
 & (W59,(EIT_C_912_1,C_CSS_V2_4)), (W60,(C_CSS_V2_4,EIT_C_CSS_R)), \\
 & (W61,(P10,EPA_C_911_1)), (W62,(EPA_C_911_1,EPA_C_912_1)), \\
 & (W63,(EPA_C_912_1,EPA_C_CSS_2)), (W64,(EPA_C_CSS_2,J9)), \\
 & (W65,(J9,C_CSS_V2_R)), (W66,(J9,C_CSS_V2_L)), (W67,(C_CSS_V2_R,L_ITCS_1)), \\
 & (W68,(C_CSS_V2_L,L_ITCS_2)), (W69,(L_ITCS_1,J10)), (W70,(J10,L_ITCS_B)), \\
 & (W71,(L_ITCS_2,J10)), (W72,(L_ITCS_B,N7))\}
 \end{aligned}$$

The components of the *Components* set must be differentiated in order to support the be-

havioural description of the system. Each component has a function in the relay-based RIS, so it is important to divide this set as a way to define the type of each component. The differentiation of the components from the ITCS example based on the relay diagram is defined as:

$$\begin{aligned}
PosSources &= \{P1, P2, P3, P4, P5, P6, P7, P8, P9, P10\} \wedge \\
NegSources &= \{N1, N2, N3, N4, N5, N6, N7\} \wedge \\
Buttons &= \{L_C_CSS_B, L_ITCS_B\} \wedge Levers = \{L_C_CSS, L_ITCS\} \wedge \\
LeverContacts &= \{L_C_CSS_1, L_C_CSS_2, L_C_CSS_3, L_ITCS_1, L_ITCS_2\} \wedge \\
MonostableContacts &= \{KIT_C_CSS_1, KAU_V2_1, SS_E_V2_1, SS_E_V2_2, \\
&SS_E_V2_3, EPA_C_911_1, EPA_C_912_1, EPA_C_CSS_1, EPA_C_CSS_2, \\
&KAG_a_G_1, RPD_FA_C911_1, INT_AC_V2_1, INT_AC_V2_2, INT_AC_V2_3, \\
&INT_AC_V2_4, PG_911_1, PG_911_2, PG_911_3, PG_911_4, KSS_E_V2_1\} \wedge \\
BistableContacts &= \{EIT_C_912_1, C_CSS_V2_1, C_CSS_V2_2, C_CSS_V2_3, \\
&C_CSS_V2_4, EIT_C_CSS_1, EIT_C_CSS_2, EIT_C_CSS_3, EIT_C_CSS_4\} \wedge \\
Junctions &= \{J1, J2, J3, J4, J5, J6, J7, J8, J9, J10\} \wedge \\
Blocks &= \{TA_SS_E_V2\} \wedge Outputs = \{EF11\} \wedge \\
MonostableRelays &= \{KIT_C_CSS, KAU_V2, SS_E_V2, EPA_C_911, EPA_C_912, \\
&EPA_C_CSS, KAG_a_G, RPD_FA_C911_1, INT_AC_V2, PG_911, KSS_E_V2, \\
&KIT_C_911\} \wedge \\
BistableRelays &= \{EIT_C_912, C_CSS_V2, EIT_C_CSS\} \wedge \\
BistableCoils &= \{C_CSS_V2_R, C_CSS_V2_L, EIT_C_CSS_R, EIT_C_CSS_L\} \wedge \\
Capacitors &= \{ \} \wedge CapPlates = \{ \} \wedge Resistors = \{ \}
\end{aligned}$$

Then, following the structural model, one may then define the levers structure by describing the relation between the levers and their contacts. Furthermore, one must define the structural link between the levers and contacts states. This structural definition is important in order to define the behaviour of these components. The specialisation of this part of the structural model for the ITCS example is defined as:

$$\begin{aligned}
LeverContactsRel_{\rightarrow} &= \{(L_C_CSS_1, L_C_CSS), (L_C_CSS_2, L_C_CSS), \\
&(L_C_CSS_3, L_C_CSS), (L_ITCS_1, L_ITCS), (L_ITCS_2, L_ITCS)\} \wedge \\
ConfigRel_{\leftrightarrow} &= \\
&\{((L_C_CSS, config_a), (L_C_CSS_1, false)), ((L_C_CSS, config_a), (L_C_CSS_2, false)), \\
&((L_C_CSS, config_a), (L_C_CSS_3, true)), ((L_C_CSS, config_b), (L_C_CSS_1, true)), \\
&((L_C_CSS, config_b), (L_C_CSS_2, true)), ((L_C_CSS, config_b), (L_C_CSS_3, false)), \\
&((L_ITCS, config_a), (L_ITCS_1, false)), ((L_ITCS, config_a), (L_ITCS_2, true)), \\
&((L_ITCS, config_b), (L_ITCS_1, true)), ((L_ITCS, config_b), (L_ITCS_2, false))\}
\end{aligned}$$

The relays are the most important part of the structural and behavioural model since they are the responsible for opening and closing the contacts. Thus, this component is responsible for controlling the flux of electrical current inside the wires. Based in the general model, in this specialisation, one must define the relays structure, connections and their relations with the contacts:

$$\begin{aligned}
MonoContactsConn_{\rightarrow} &= \{((KIT_C_CSS_1, up), W39), ((KAU_V2_1, up), W40), \\
&((SS_E_V2_1, down), W2), ((SS_E_V2_2, up), W25), ((SS_E_V2_3, up), W46), \\
&((EPA_C_911_1, up), W62), ((EPA_C_912_1, up), W63), ((EPA_C_CSS_1, up), W57), \\
&((EPA_C_CSS_2, up), W64), ((KAG_a_G_1, up), W9), ((RPD_FA_C911_1, up), W11), \\
&((KSS_E_V2_1, up), W28), ((INT_AC_V2_1, up), W5), ((INT_AC_V2_2, down), W15), \\
&((INT_AC_V2_3, down), W34), ((INT_AC_V2_4, up), W48), ((PG_911_1, up), W6), \\
&((PG_911_2, up), W13), ((PG_911_3, down), W32), ((PG_911_4, up), W42)\} \wedge \\
BistContactsConn_{\rightarrow} &= \{((EIT_C_912_1, left), W59), ((C_CSS_V2_1, right), W3), \\
&((C_CSS_V2_2, right), W24), ((C_CSS_V2_3, right), W42), ((C_CSS_V2_4, right), W60), \\
&((EIT_C_CSS_1, left), W3), ((EIT_C_CSS_2, right), W28), ((EIT_C_CSS_3, right), W31), \\
&((EIT_C_CSS_4, right), W48)\} \wedge \\
BistableCoilsRel_{\rightarrow} &= \{(C_CSS_V2_R, C_CSS_V2), (C_CSS_V2_L, C_CSS_V2), \\
&(EIT_C_CSS_R, EIT_C_CSS), (EIT_C_CSS_L, EIT_C_CSS)\} \wedge \\
CoilSide_{\rightarrow} &= \{(C_CSS_V2_R, right), (C_CSS_V2_L, left), (EIT_C_CSS_R, right), \\
&(EIT_C_CSS_L, left)\} \wedge \\
MonoRelayContactsRel_{\rightarrow} &= \{(KIT_C_CSS_1, KIT_C_CSS), (KAU_V2_1, KAU_V2), \\
&(SS_E_V2_1, SS_E_V2), (SS_E_V2_2, SS_E_V2), (SS_E_V2_3, SS_E_V2), \\
&(EPA_C_911_1, EPA_C_911), (EPA_C_912_1, EPA_C_912), \\
&(EPA_C_CSS_1, EPA_C_CSS), (EPA_C_CSS_2, EPA_C_CSS), (KAG_a_G_1, KAG_a_G), \\
&(RPD_FA_C911_1, RPD_FA_C911), (INT_AC_V2_1, INT_AC_V2), \\
&(INT_AC_V2_2, INT_AC_V2), (INT_AC_V2_3, INT_AC_V2), \\
&(INT_AC_V2_4, INT_AC_V2), (PG_911_1, PG_911), (PG_911_2, PG_911), \\
&(PG_911_3, PG_911), (PG_911_4, PG_911), (KSS_E_V2_1, KSS_E_V2)\} \wedge \\
BistRelayContactsRel_{\rightarrow} &= \{((EIT_C_912_1, EIT_C_912), (C_CSS_V2_1, C_CSS_V2), \\
&(C_CSS_V2_2, C_CSS_V2), (C_CSS_V2_3, C_CSS_V2), (C_CSS_V2_4, C_CSS_V2), \\
&(EIT_C_CSS_1, EIT_C_CSS), (EIT_C_CSS_2, EIT_C_CSS), \\
&(EIT_C_CSS_3, EIT_C_CSS), (EIT_C_CSS_4, EIT_C_CSS)\}.
\end{aligned}$$

As the ITCS example contains a block, one must also formalise the structure of this component. As presented in the general model, the specialisation must define the block type, and their

connections. This is the last part of the structural formalisation. The ITCS case study block specialisation is defined as:

$$\begin{aligned} BlockTypes_{\rightarrow} &= \{(TA_SS_E_V2, timed_act)\} \wedge \\ BlockDepConn_{\leftrightarrow} &= \{(TA_SS_E_V2, SS_E_V2)\} \wedge \\ BlockIndConn_{\leftrightarrow} &= \{(TA_SS_E_V2, C_CSS_V2_2), (TA_SS_E_V2, J5)\}. \end{aligned}$$

This structural formalisation contains all the relay diagram information that is required in order to be able to define the system behaviour. Based on the mathematical basis used for the structural definition, it is possible to describe the system behaviour by defining the relation between the specified electrical components states, as presented in the next section.

3.5.2 Behavioural Formalisation and Verification

The system behaviour depends on the system structure, which generally varies depending on the system context. However, by considering that each component type has a unique invariable behaviour, it is possible to generalise the behavioural logic based on these fixed behaviours, as presented in the behavioural definitions of this chapter. So, for the ITCS example, as well as for many others case studies, the behavioural formalisation is the same general description composed by the behavioural definitions presented in this chapter.

In this context, the general behavioural formalisation of the ITCS case study must initially contain the behavioural definition for the system contacts and inputs. The description of the inputs is extremely important in order to define and analyse the system state as these components are responsible for causing the system instability and state evolution. Moreover, the contacts and inputs states are essential to define whether some components are electrified or not, since these components may block the electric current flow. The formalisation of the buttons, levers and contacts states are defined as:

$$\begin{aligned} MonoContactsSt_{\rightarrow} &\in (MonostableContacts \rightarrow U_D) \wedge \\ BistContactsSt_{\rightarrow} &\in (BistableContacts \rightarrow R_L) \wedge \\ ButtonsSt_{\rightarrow} &\in (Buttons \rightarrow \mathbb{B}) \wedge \\ LeverSt_{\rightarrow} &\in (Levers \rightarrow LeverConfig) \wedge \\ LeverContactsSt_{\rightarrow} &\in (LeverContacts \rightarrow \mathbb{B}) \wedge \\ LeverContactsSt_{\rightarrow} &= ConfigRel_{\leftrightarrow}[LeverSt_{\rightarrow}] \wedge \end{aligned}$$

The contacts and the inputs of the system are responsible for part of the system electrification as they may allow or deny the components electrification. These components may block the passage of electrical current as they may disconnect from certain wires. Thus, in order to define if the components are electrified or not in the ITCS example, it is important to define these components disconnections as well as a new incidence function that considers the system

disconnections. These definitions are described as:

$$\begin{aligned}
Disconnections &= (MonoContactsConn_{\leftrightarrow}[MonoContactsSt_{\rightarrow}] \cup \\
&BistContactsConn_{\leftrightarrow}[BistContactsSt_{\rightarrow}] \cup \\
&(Incidence_{\rightarrow}^{-1}[ran(Incidence_{\rightarrow}) \triangleright (LeverContactsSt_{\rightarrow}^{-1}[\{false\}] \cup ButtonsSt_{\rightarrow}^{-1}[\{false\})]) \cup \\
&(Incidence_{\rightarrow}^{-1}[(LeverContactsSt_{\rightarrow}^{-1}[\{false\}] \cup ButtonsSt_{\rightarrow}^{-1}[\{false\})] \triangleleft ran(Incidence_{\rightarrow})]) \wedge \\
RC_{\leftrightarrow} &= (Disconnections \triangleleft Incidence_{\rightarrow})
\end{aligned}$$

However, the ITCS example contains a block in a way that the components may also be electrified when connected to this component. So, one must also use the block formalisation in order to define the components electrification condition. These definitions are described as:

$$\begin{aligned}
BlockPassOfTime_{\rightarrow} &\in (Blocks \rightarrow \mathbb{B}) \wedge BlockIsElectrified_{\rightarrow} \in (Blocks \rightarrow \mathbb{B}) \wedge \\
BlockIsActivated_{\rightarrow} &\in (Blocks \rightarrow \mathbb{B}) \wedge
\end{aligned}$$

$$\begin{aligned}
BlockIsElectrified_{\rightarrow}(bl) &\Leftrightarrow \\
&\exists es, Pa_{\rightarrow}.(es \in (PosSources \cup NegSources) \wedge \\
&(((Pa_{\rightarrow} \in path(bl, es, RC_{\leftrightarrow}) \wedge (bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
&(Pa_{\rightarrow} \in cycle(bl, RC_{\leftrightarrow}) \wedge ((bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow}) \wedge \\
&((bl, last(front(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})))) \wedge
\end{aligned}$$

$$\begin{aligned}
BlockIsActivated_{\rightarrow}(bl) &\Leftrightarrow \\
&((BlockTypes_{\rightarrow}(bl) = timed_act \wedge BlockIsElectrified_{\rightarrow}(bl) \wedge PassOfTime_{\rightarrow}(bl)) \vee \\
&(BlockTypes_{\rightarrow}(bl) = timed_deact \wedge BlockIsElectrified_{\rightarrow}(bl)) \vee \\
&(BlockTypes_{\rightarrow}(bl) = timed_deact \wedge \neg BlockIsElectrified_{\rightarrow}(bl) \wedge \neg PassOfTime_{\rightarrow}(bl)) \wedge
\end{aligned}$$

$$\begin{aligned}
electrified(comp) &\Leftrightarrow \\
&((\exists ps, ns, Pa_{\rightarrow}.(ps \in PosSources \wedge ns \in NegSources \wedge \\
&Pa_{\rightarrow} \in path(ps, ns, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}))) \vee \\
&(\exists bl, es, Pa_{\rightarrow}.(bl \in blocks \wedge es \in (PosSources \cup NegSources) \wedge \\
&Pa_{\rightarrow} \in path(bl, es, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge (bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
&(\exists bl, Pa_{\rightarrow}.(bl \in blocks \wedge Pa_{\rightarrow} \in cycle(bl, RC_{\leftrightarrow}) \wedge comp \in ran(Pa_{\rightarrow}) \wedge \\
&(bl, first(tail(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow} \wedge (bl, last(front(Pa_{\rightarrow}))) \in BlockIndConn_{\leftrightarrow})) \vee \\
&(\exists bl, Pa_{\rightarrow}.(bl \in blocks \wedge Pa_{\rightarrow} \in cycle(bl, RC_{\leftrightarrow}) \wedge \\
&comp \in ran(Pa_{\rightarrow}) \wedge BlockIsActivated_{\rightarrow}(bl) \wedge (bl, first(tail(Pa_{\rightarrow}))) \in BlockDepConn_{\leftrightarrow} \wedge \\
&(bl, last(front(Pa_{\rightarrow}))) \in BlockDepConn_{\leftrightarrow})))
\end{aligned}$$

Once the components electrification condition is defined, one may detail the conditions for the relays to be activated as well as the relation between these components and the contacts:

$$\text{MonoRelaysSt}_{\rightarrow} \in (\text{MonostableRelays} \rightarrow \mathbb{B})$$

$$\text{BistRelaysSt}_{\rightarrow} \in (\text{BistableRelays} \rightarrow R_L)$$

$$\text{RelayPrevSt}_{\rightarrow} \in \text{BistableRelays} \rightarrow L_R$$

$$\forall mr.(mr \in \text{MonostableRelays} \Rightarrow$$

$$(\text{MonoRelaysSt}_{\rightarrow}(mr) \Leftrightarrow \text{electrified}(mr)))$$

$$\forall br, c1, c2.((br \in \text{BistableRelays} \wedge$$

$$c1 \in \text{BistableCoils} \wedge c2 \in \text{BistableCoils} \wedge$$

$$\text{BistableCoilsRel}_{\rightarrow}^{-1}[\{br\}] = \{c1, c2\}) \Rightarrow$$

$$(((\text{electrified}(c1) \wedge \neg \text{electrified}(c2)) \Rightarrow \text{BistRelaysSt}_{\rightarrow}(br) = \text{CoilSide}_{\rightarrow}(c1)) \wedge$$

$$((\neg \text{electrified}(c1) \wedge \text{electrified}(c2)) \Rightarrow \text{BistRelaysSt}_{\rightarrow}(br) = \text{CoilSide}_{\rightarrow}(c2)) \wedge$$

$$((\neg \text{electrified}(c1) \wedge \neg \text{electrified}(c2)) \Rightarrow \text{BistRelaysSt}_{\rightarrow}(br) = \text{RelayPrevSt}_{\rightarrow}(br)) \wedge$$

$$((\text{electrified}(c1) \wedge \text{electrified}(c2)) \Rightarrow \text{BistRelaysSt}_{\rightarrow}(br) = \text{RelayPrevSt}_{\rightarrow}(br))))$$

$$\forall mc.(mc \in \text{MonostableContacts} \Rightarrow$$

$$((\text{MonoRelaysSt}_{\rightarrow}(\text{MonoRelayContactsRel}_{\rightarrow}(mc)) \Rightarrow \text{MonoContactsSt}_{\rightarrow}(mc) = \text{up}) \wedge$$

$$(\neg \text{MonoRelaysSt}_{\rightarrow}(\text{MonoRelayContactsRel}_{\rightarrow}(mc)) \Rightarrow \text{MonoContactsSt}_{\rightarrow}(mc) = \text{down})))$$

$$\forall bc.(bc \in \text{BistableContacts} \Rightarrow$$

$$(\text{BistContactsSt}_{\rightarrow}(bc) = \text{BistRelaysSt}_{\rightarrow}(\text{BistRelayContactsRel}_{\rightarrow}(bc))))$$

This last expression finishes the general behavioural formalisation, which may be used as basis for the safety verification of the system. As the components states are logically related, it is possible to determine which states may never occur at the same time as a way to avoid dangerous situations. In the ITCS example, for instance, one may guarantee that the components *KIT_C_911* and *EF11* must never be activated at the same time, preventing the signals close to the shared portion of the tracks to be opened at the same time. Thus, this safety condition avoids the occurrence of collisions in this shared tracks by allowing only one train to pass at a time. So, for every combination of inputs states (for the input components and the time variables), $\text{electrified}(\text{KIT_C_911}) \wedge \text{electrified}(\text{EF11})$ must never be true, which may be defined as:

$$\neg(\text{electrified}(\text{KIT_C_911}) \wedge \text{electrified}(\text{EF11}))$$

In order to prove the system safety, one may assure that this expression is always true. So, considering the paths that may lead the components to be activated (which is the basis of the electrification condition), this expression may be rewritten as a way to be based on the contacts states. This is because the paths are defined according to the contacts and inputs states that may

allow the current to flow. So this expression may be rewritten as:

$$\begin{aligned} &\neg((MonoContactsSt_{\rightarrow}(KIT_C_CSS_1) = up \wedge MonoContactsSt_{\rightarrow}(KAU_V2_1) = up \wedge \\ &\quad BistContactsSt_{\rightarrow}(C_CSS_V2_3) = right \wedge MonoContactsSt_{\rightarrow}(PG_911_4) = up) \\ &\wedge \\ &\quad (MonoContactsSt_{\rightarrow}(SS_E_V2_1) = down \wedge BistContactsSt_{\rightarrow}(C_CSS_V2_1) = right \wedge \\ &\quad BistContactsSt_{\rightarrow}(EIT_C_CSS_1) = left \wedge MonoContactsSt_{\rightarrow}(INT_AC_V2_1) = up \wedge \\ &\quad MonoContactsSt_{\rightarrow}(PG_911) = up_1)); \end{aligned}$$

i.e., the precondition for KIT_C_911 to be activated may never be true in the same state that the precondition for $EF11$ to be activated is also true. Then, as the contact state is a result of the relays states, it is possible to rewrite this expression based solely on the relays states:

$$\begin{aligned} &\neg((MonoRelaysSt_{\rightarrow}(KIT_C_CSS) = true \wedge MonoRelaysSt_{\rightarrow}(KAU_V2) = true \wedge \\ &\quad BistRelaysSt_{\rightarrow}(C_CSS_V2) = right \wedge MonoRelaysSt_{\rightarrow}(PG_911) = true) \\ &\wedge \\ &\quad (MonoRelaysSt_{\rightarrow}(SS_E_V2) = false \wedge BistRelaysSt_{\rightarrow}(C_CSS_V2) = right \wedge \\ &\quad BistRelaysSt_{\rightarrow}(EIT_C_CSS) = left \wedge MonoRelaysSt_{\rightarrow}(INT_AC_V2) = true \wedge \\ &\quad MonoRelaysSt_{\rightarrow}(PG_911) = true)) \end{aligned}$$

As the component KIT_C_CSS depends on others contacts to be activated, one may rewrite this expression by considering the contacts and relays that lead to the activation of this component:

$$\begin{aligned} &\neg(((LeverContactsSt_{\rightarrow}(L_C_CSS_1) = true \wedge MonoRelaysSt_{\rightarrow}(SS_E_V2) = true \wedge \\ &\quad BistRelaysSt_{\rightarrow}(EIT_C_CSS) = right \wedge MonoRelaysSt_{\rightarrow}(INT_AC_V2) = true) \wedge \\ &\quad MonoRelaysSt_{\rightarrow}(KAU_V2) = true \wedge BistRelaysSt_{\rightarrow}(C_CSS_V2) = right \wedge \\ &\quad MonoRelaysSt_{\rightarrow}(PG_911) = true) \\ &\wedge \\ &\quad (MonoRelaysSt_{\rightarrow}(SS_E_V2) = false \wedge BistRelaysSt_{\rightarrow}(C_CSS_V2) = right \wedge \\ &\quad BistRelaysSt_{\rightarrow}(EIT_C_CSS) = left \wedge MonoRelaysSt_{\rightarrow}(INT_AC_V2) = true \wedge \\ &\quad MonoRelaysSt_{\rightarrow}(PG_911) = true)) \end{aligned}$$

So, based on this expression, one may conclude that, one of the conditions for the components KIT_C_911 and $EF11$ to be activated is that the component EIT_C_CSS is in the right and left states at the same time, which is not possible. Because of this contradiction, the expression $electrified(KIT_C_911) \wedge electrified(EF11)$ can be completely falsified and its negation is always considered *true*. So, it is possible to assure that the logic of this system may never allow these two components to be activated at the same time.

The other verification that may be made in this ITCS example is regarding the possibility of derailment if the turnout is not locked. A similar logical expression may be made in order to assure that the component *KIT_C_911* is not activated at the same time that the component *KAG_a_G* is activated. This is because the signal must not be opened at the same time that the turnout is not allowing the train to change tracks. This safety measure may guarantee that, when a train pass by the green signal, the turnout may not unexpectedly change position, which could cause a derailment. Furthermore, it avoids that the train goes to the blocked tracks, which is the intent of the ITCS system. However, the analysis of the RIS formal model cannot prove the safety of the system as the component *KAG_a_G* is an input on the ITCS relay diagram. As a consequence, according to the formalisation logic, the turnout may change its position at anytime, regardless of the rest of the system, as it is controlled by the environment. Nonetheless, more details about this component is presented in other relay diagrams as a way that the system safety may still be analysed. By any means, SNCF states that the relation between these components are well defined in order to lock the turnout when a train needs to pass over it.

As the relay diagrams are not complete, it is not possible to extract all the information required for assuring the system safety. One solution for acquiring all the information needed, is the enrichment of these diagrams with a conceptual model of the relation between the components. By using a conceptual model based on a well grounded ontology, one may be able to model the knowledge about the system context and environment. This model may be used as basis for the improvement of the system formalisation and formal specification as the knowledge about the system may be considered. The use of conceptual modelling in this context is a future work as it is detailed in the conclusion of this thesis.

3.6 Discussion

The structural mathematical model presented in this chapter is a transcription of the structural information depicted in the relay diagram. The physical relation between components and wires are represented in a graph format. Besides, all other structural relations are represented as mathematical relations and functions supported by the Set Theory. Some examples of these relations are the magnetic influence between the relays and contacts and the physical connection between a lever and its contacts. Thus, the mathematically formalised structural model represent the same information depicted in the relay diagram drawing.

However, the system behaviour is not described in the relay diagram. The only documentation about the system execution is the general description of each component behaviour in natural language. In order to support the formal specification and verification of the RIS, this chapter presents a first formalisation of the behaviour of each component by relating their possible states with the states of other components in the same system. It is important to note that this behavioural model do not present the system state succession or the system proper execution. In fact, the behavioural model describes all the information we can obtain from the structural relation between the components, their possible states and the conditions for these components to assume each of their states according to what can be found in literature. As a result, we have obtained a general behavioural model that can basis any system analysis and formal specification.

In this context, the correctness of the formalised model depends on the precise transcription

of the diagram information into the mathematical model. The well-definedness properties presented in this chapter provide means for guaranteeing that the modelled structure does not contain impractical relations between the components (like a bistable relay containing more than two coils, for instance). Nonetheless, in order to guarantee the correctness of the structural model, one must assure that every relation between two components is represented in the mathematical model accordingly to what is presented in the relay diagram. Although logic can be used in order to verify that the system is well defined according to the well-definedness properties, it cannot be used for guaranteeing that it contains all the correct components relations, as the transcription is made from a completely graphical model to a mathematical one.

The precise transcription of the system structure is also important in order to guarantee the behavioural correctness. As the behavioural model does not change according to the case study, it depends exclusively on the relations presented in the structural model. An error in the transcription can impact on the verification of the safety conditions, ruining the system verification.

Although this work provides all the structural definitions that can guide the model transcription, it cannot guarantee the absence of human errors as the process is made completely manually. A possible solution to this problem is the creation of a tool to automate the transcription of the information presented in the relay diagram to the structural model according to the structural definitions presented in this chapter. Furthermore, SAT-solvers can also be applied in order to guarantee that the modelled system meets the established well-definedness properties and safety conditions. The creation of this tool is a perspective of this work. In this thesis, we are concerned on providing all the conceptual basis for modelling the relay-based RIS structure and behaviour.

3.7 Formal Specification Based on the Formalisation

The formalisation presented in this chapter details how Logic, Set Theory and other mathematical foundations may be used in order to describe the structure and behaviour of relay-based Railway Interlocking Systems. However, this formalisation has a tendency to contain complex and extensive logical expressions, making it difficult to manually defining and proving the systems. Although it may be considered as a reasonable solution for the ancient relay diagrams manual analysis based on interpretation, it is possible to make use of the many existing modern tools in order to support the definition, analysis, proof and even simplification of the system models.

In this context, the formalisation presented in this chapter may be interpreted as an important step towards the system formal specification as it gives a strong logical interpretation of the information contained inside the relay diagrams. So, the logic for the system structure and behaviour presented in this chapter may be used as basis for the system formal specification in many different languages, like B, Z or Petri Nets. In this context, the mathematical foundations used for this formalisation are the basis for many formal methods, which may ease the process of adapting this formalisation to the syntax and semantics of a formal specification language. One important example of such formalism is the B-method, which is a formal method that has been successfully used in the railway field.

In the next chapter, it is presented how the formalisation of the RIS structure and behaviour may be interpreted and adapted in order to allow the formal specification of the information

contained inside the relay diagrams. As these diagrams generally only contain structural information, the behavioural formalisation presented in this chapter is a strong foundation in order to define the system behaviour and then verify its safety. These transformation directives are the first step towards automating the transformation from the diagrams to a formal specification as they give some guidelines on how to perform this transformation. Moreover, based on the B-method input definition and state succession support, it is possible to simplify the logic presented in this chapter and still improve the system description by the use of the B-method modern notations.

RIS B Formal Specification: A Diagram-specific Approach

Contents

4.1 Introduction	118
4.2 Behavioural Specification Based on the System State Space	119
4.2.1 System Variables and State-space Organisation	119
4.2.2 State Evolution Specification	124
4.3 Discussion	136

4.1 Introduction

The use of Formal Methods in the railway industry is a strongly recommended practice as it allows the proof of the system safety. The formal specification languages mathematical background and supporting tools may allow not only the specification, analysis and verification of these systems, but also the refinement and implementation. These latter features can support the generation of safety-proved computer-based systems, which can be useful to the railway industry in order to deal with the safety-critical aspects of this field. The use of a formal software development process can be the differential factor in order to guarantee the system safety in many situations.

In the SNCF signalisation context, it is possible to apply Formal Methods in order to specify the information that can be captured from the relay diagrams as a way to prove the system safety and transform these systems into computer-based ones. Nonetheless, it is important to consider the fact that these diagrams describe only the structure of the electrical circuits as a way that it is necessary to make a deep analysis of the system behaviour in order to examine its safety.

In this context, the formalisation of the relay diagrams information presented in Chapter 3 provides a structural and behavioural logical representation of the system that is able to support the behavioural formal specification of the relay-based Railway Interlocking Systems. Although this formalisation already offers the possibility of proving system properties, it is true that the manual specification and proof of the logical expressions may be difficult, time consuming and error prone. So, the use of a formal language to support the specification and proof of these systems is important in order to make this approach industrially beneficial.

Many different formal specification languages can be used in order to specify Railway Interlocking Systems. As the relay-based RIS formalisation presented in the last chapter is grounded on the same mathematical foundation of many of these languages, the logical expressions can be adapted to many different formalisms, like B, CSP, Petri Nets or Z. In that respect, B is a strong language for the specification of these systems due to its successful history in the railway field, well documented syntax, strong mathematical foundation and the existence of many supporting tools that allow the specification, analysis, proof and refinement of the systems.

Aiming at the specification of the relay-based RIS behaviour, this chapter presents an approach to adapt the relay-based RIS structural and behavioural formalisation in order to formally specify these systems in B. This adaptation is based on the logic used and the transformation of the expressions in order to adjust to the advantages and limitations of the formal language. In this context, B disposes of many functionalities that may be used in order to enhance the formalisation by considering other aspects, like the state succession and the input handling, for instance.

However, as the formalisation presented in the last chapter disposes of complex logical expressions and structures derived from relations, a reformulation of these expressions must be considered in the formal specification. This is because the complex expressions require an intense analysis that demands too much effort from the supporting tools. In order to perform more efficient verifications, instead of generalising the system structure and behaviour, a diagram-specific approach for the simplification of the formalisation is used. So, in this formal specification, the structure of the system is abstracted into the behavioural logic. In this context, although the system structure is not specified, it is an essential basis for the system behavioural

specification. The reason for this is to focus on the behavioural description in order to be able to prove the safety and implement these systems as computer based RIS. In this chapter, the formal specification of the system based on the relay-based RIS formalisation is explained using the ITCS case study as a running example.

4.2 Behavioural Specification Based on the System State Space

The B-method focuses on the specification of the system state representation and evolution. The system state is determined by the combination of each variable state, so one may consider these variables as the core of the whole specification. In this context, a B-machine can be divided into two parts: (1) the system variables and state-space organisation, and (2) the state evolution specification. The former part is related to the definition of variables and their types, initial values and properties. The latter part is defined by the OPERATIONS clause and describes how the system states are changed according to the given inputs.

4.2.1 System Variables and State-space Organisation

The transformation of the logic described in the relay-based RIS formalisation to the B-method must consider all the characteristics of the language syntax and semantics. Although the B-method supports all the mathematical foundations that basis the RIS formalisation, this formal language has a basic structure that must be respected. Thus, the transformation between these models is based on the definition of transformation directives from the structural and behavioural definitions presented in the last chapter to the formal specification of each of the B-machine clauses.

While in the RIS formalisation the system state is determined by the components relations with specific values, the central focus of the B-method specification is on the variables, whose possible combination of values define the system state space. So, modelling the electrical components as B-variables is a natural modelling decision that grounds the rest of the specification. As the states of these variables are defined according to the values they may assume, it is possible to use the components states representation definitions of the RIS formalisation in order to define the variables types. Nonetheless, it is important to remember that the B-method makes an explicit differentiation between internal components and inputs that must be taken into consideration. So, as the electrical components defined in the Structural Definition SD1 does not explicitly make this differentiation, it is important to define a transformation directive for obtaining the list of variables from the list of components.

Moreover, this transformation directive must also take into consideration that the formalisation presented in the last chapter also contains several components that can be abstracted either because they have no behaviour or because they have been only created to support the system behavioural model. In the first case, junctions, resistors, positive and negative energy sources have no defined states in the RIS formalisation, thus, they do not have a defined behaviour and are not specified as variables in this specification. In the second case, lever contacts, bistable relays coils, capacitors plates, monostable and bistable contacts are components that were created to support the system behavioural model. Each of these components is part of another one in a way that their states are linked, but they have been structurally modelled as different components with

the objective of simplifying the behavioural description. As this approach for the RIS formal specification is focused on the system behaviour in a way that the structure is abstracted, there is no need on separating these components. Furthermore, by decreasing the number of variables, the size of the state space also decreases, limiting the problem of the state space explosion. As a result, the automatic verification of the formal specification with the supporting tools tends to have a better performance. The approach for determining the B-machine variables is presented in the Transformation Directive TD1.

Transformation Directive TD1 (Variables Identification): The variables for the relay-based RIS formal specification are the components defined in the Structural Definition SD1 which have defined states and which are not contacts, subcomponents or inputs. In this context, the inputs are considered as levers, buttons or relays whose coils are not part of the graph represented in the incidence function of this same definition.

Although monostable and bistable contacts are not presented as subcomponents in the RIS formalisation, a contact is essentially part of the relay, as defined in Chapter 1 of this present thesis. The state of these two components are also completely linked, as presented in the Behavioural Definitions BD12 and BD13. Furthermore, it is important to mention that, although the abstracted components are not represented as variables in this formal specification, they have an importance in the system state succession definition. Figure 4.1 presents how each component of the relay-based RIS formalisation is specified in B-method. As depicted in this figure, some components can be specified as variables or inputs. Regarding relays, they must be considered as inputs when their coils activation is controlled by external factors. On the other hand, blocks and capacitors are influenced by time, which is part of the environment in this approach. In this context, the timed aspect of these components must be taken into consideration as inputs of the system. The analysis of these components and the inputs formal specification is discussed later in this chapter.

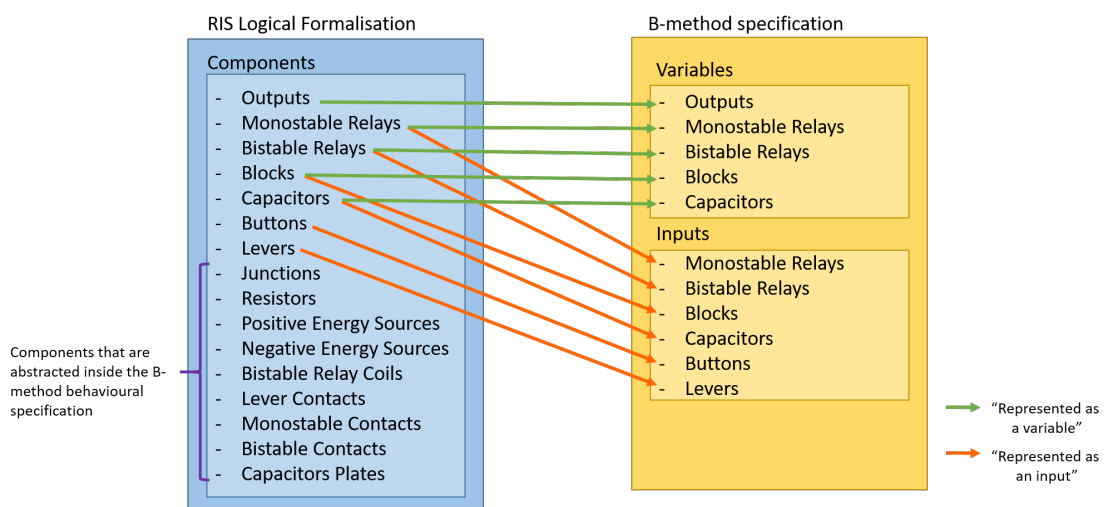


Figure 4.1 – Representation of the components in the RIS logical formalisation and in the B-method formal specification.

The VARIABLES clause of the ITCS case study is specified in the B-method as:

```
VARIABLES
KIT_C_CSS, SS_E_V2, TA_SS_E_V2, EIT_C_CSS,
C_CSS_V2, PG_911, EF11, KIT_C_911.
```

In this case, only the outputs, internal monostable and bistable relays and the system block are represented as variables. The inputs of these systems are defined later in the state succession transformation directives.

In B, the variables values are defined accordingly to the types given to them. These types are defined by sets, like the Boolean set `BOOL` that represents the set \mathbb{B} , containing the elements `TRUE` and `FALSE`. Other specific sets may be defined inside the B `SETS` clause. The definition of the components types is made inside the `INVARIANT` clause. The variables typing is not only syntactically mandatory, but it also has an importance on the definition of the system state space. As in the RIS formalisation the states of the components are defined by the relation between the components and values sets, the transformation between the models can be simply grounded on the state representation behavioural definitions.

Nonetheless, we propose a single important change: the creation of a model less generalised and more focused on a specific context. In this case, instead of using general states definition like *right* and *left*, we propose the use of the states described in the relay diagram itself. This formal specification is the specialisation of the more general behavioural model described in the formalisation, which is completely objective and unconcerned with the system context (the case study structure). By using the system specific information, it is possible to enhance the specification readability by using the diagram as support. The existence of these specific components states are discussed in Chapter 1 of this present thesis. In the ITCS example, for instance, the bistable relay `C_CSS_V2`, depicted in Figure 4.2, may assume the states `DV` and `ES`, which represent, respectively, the *left* and *right* states defined in the RIS formalisation. So, instead of defining a set `{left, right}` for typing this bistable relay, in our approach one must type it according to the set `{DV, ES}`. The formal specification of the variables typing is defined in the Transformation Directive TD2. A mapping between the state representation in the RIS formalisation and in the formal specification is presented in Table 4.1.

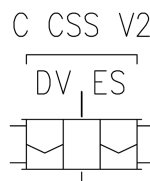


Figure 4.2 – Relay `C_CSS_V2` of the ITCS case study representing the states that this component may assume.

Transformation Directive TD2 (Variables Typing): The type of each component is defined as presented in the formalisation behavioural definitions. For monostable relays, outputs, blocks and capacitors, the activation of these components is represented by a Boolean value, as described in the Behavioural Definitions `BD8`, `BD14`, `BD17` and `BD21`, respectively. The type of bistable relays, however, is defined accordingly to the Behavioural Definition `BD9`, but the *left* and *right*

states are replaced by their respective positions established in the relay diagram.

Table 4.1 – Components state representation mapping.

Component	RIS formalisation	Formal specification
Monostable relays	Boolean	Boolean
Bistable relay	<i>right or left</i>	Specific representation inside the relay diagram
Output	Boolean	Boolean
Blocks	Boolean	Boolean
Capacitors	Boolean	Boolean

Based on this transformation directive, in the ITCS case study, it is possible to define the SETS clause:

SETS

```
O_OR_F = {POS_0, POS_F};
DV_OR_ES = {POS_ES, POS_DV}.
```

These sets support the specification of the system bistable relays states. The rest of the components states are defined according to the Boolean set B00L. The ITCS case study INVARIANT clause can be then defined as:

INVARIANT

```
KIT_C_CSS : B00L &
SS_E_V2 : B00L &
TA_SS_E_V2 : B00L &
EIT_C_CSS : O_OR_F &
C_CSS_V2 : DV_OR_ES &
PG_911 : B00L &
EF11 : B00L &
KIT_C_911 : B00L
```

The INVARIANT clause has the objective of determining system properties that must be always met during the system execution. This is why the variables typing is defined inside this clause. The system verification analyses if there is the possibility of a state to violate the properties defined inside the invariant. If it detects that a state succession may cause this violation, the B-machine is considered inconsistent. So, this clause may also be used in order to define safety properties, for instance. The lack of inconsistencies of a machine after a verification regarding a safety property is the first step towards the proof of the system safety. In this context, a safety property regarding a relay-based RIS behaviour is defined in B with the same Boolean logic as presented in the Section 3.4.2. The only difference in this definition is regarding the syntax used as it is necessary to adapt the formalisation to the B-method notation. The definition of the RIS safety properties in B is presented in the Transformation Directive TD3.

Transformation Directive TD3 (Safety Properties Specification): The relay-based RIS safety properties that must be verified are specified inside the B-method INVARIANT clause accordingly to the same Boolean logic defined in the formalisation, as presented in Section 3.4.2. The only adaptation of the proposed expression is regarding the syntax of the expressions.

In the ITCS case study, for instance, the safety property for avoiding frontal collisions that was defined as $\neg(\text{electrified}(\text{KIT_C_911}) \wedge \text{electrified}(\text{EF11}))$ can be completely transcribed to the B-method notations inside the INVARIANT clause as:

$$\text{not}(\text{KIT_C_911} = \text{TRUE} \ \& \ \text{EF11} = \text{TRUE}).$$

If there is a possibility of achieving a state where this condition is not met, the specification is not consistent. As a consequence, the system cannot be considered safe.



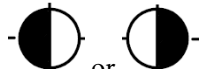

The last clause for the system variables and state-space organisation is the INITIALISATION. This is a special clause that enforces the definition of an initial value for each of the variables, so the specification can be analysed through its possible state successions. As a logic for the system state succession is not defined in the relay-based RIS formalisation, this information is not formalised, so it is not possible to use any of the structural and behavioural definitions as basis for specifying the system initialisation. Nonetheless, this information is still presented in the relay diagram, where the system is always presented in a functional existing state. Since it is a valid state, it is a perfect candidate for the system initialisation.

In this context, the components states can be obtained from the relay diagrams by the following logic:

- Monostable relays are activated if their monostable contacts are in the *up* state, going against the gravity, as the relay and contacts states are always linked;
- Bistable relays are in the *right* and *left* states if their bistable contacts are also in the *right* and *left* states, respectively, as the relay and contacts states are always linked;
- Outputs are activated if the system state allows its electrification according to the Behavioural Definition BD23 and taking into consideration the contacts and buttons states as depicted in the diagram;
- Blocks are deactivated as the passing of time cannot be predicted.

In this context, buttons and lever contacts states can also be deduced from the relay diagrams as depicted in Table 4.2, which is useful in order to determine the outputs states. As the passing of time cannot be predicted, it is not possible to determine if the block is activated or not, even if it is electrified according to the Behavioural Definition BD18. Nonetheless, it does not represent a problem as the formal specification of the system state succession may activate it in the next state since the passing of time information is indicated in the system inputs. The state succession logic is presented later in this chapter. The B-machine initialisation definition is presented in the Transformation Directive TD4.

Table 4.2 – Representation of buttons and lever contacts states inside the relay diagram.

Component	Opened	Closed
Button		
Lever Contact		

Transformation Directive TD4 (Initialisation Definition): As the relay-based RIS formalisation lacks a logic for the system state succession, it does not contain information regarding the system initialisation. Nonetheless, as the relay diagram represents a functional state, the formal specification of each component initial value can be based on their drawings inside the relay diagram.

So, based on the relay diagram drawings, one may derive the ITCS variables initial state and specify it in the INITIALISATION clause as:

```
INITIALISATION
KIT_C_CSS := FALSE ||
SS_E_V2 := FALSE ||
EIT_C_CSS := POS_F ||
C_CSS_V2 := POS_DV ||
PG_911 := TRUE ||
TA_SS_E_V2 := FALSE ||
EF11 := FALSE ||
KIT_C_911 := FALSE
```

The INITIALISATION clause is the last part of the state-space organisation of the B-machine. Regarding the inputs of the system, they are treated in the next B-clause, OPERATIONS, which focuses on the definition of the system state evolution. As the structural relation between the components is essential in order to define their states, in this next part of the specification the structural and behavioural definitions of the formalisation are used as basis in order to support the logic for the system state evolution.

4.2.2 State Evolution Specification

The operations of a B-machine specify the rules for the system state evolution. Although the behavioural formalisation of the relay-based RIS does not explicitly present the rules for the system state succession, it formalises the relation between the components states and the preconditions for their activation. So, these preconditions can be used as the basis for the specification of an operation for the description of the system state evolution.

In this context, it is important to define one unique operation for the state evolution of the whole system. As this approach does not take into account the transient states, all the components must evolve at the same time, reaching a system state after the execution of the operation. It is important to mention that the formal specification of concurrent systems (which is not supported by B-method [Korečko and Sobota 2014]) could result in a more realistic specification of the real system, as each component is physically a subsystem with its own behaviour. Nonetheless, the B-method is still able to reproduce the complete system states based on the relation between the inputs and the components behaviour, which is the most important feature in order to implement these systems as computer-based Railway Interlocking Systems.

The operation for the system state evolution must initially receive the inputs, which are responsible for triggering the system to evolve. The list of inputs must contain all the components that act like part of the interface between the system and the environment. These components are buttons, levers and external relays, which were presented in the Structural Definitions SD1

and SD2. The external relays, in this case, are the ones that are represented in the components set and which cannot be treated as a variable of the B-machine. Furthermore, as the time is considered in this approach as an environmental aspect, as presented in the Behavioural Definitions BD16 and BD20, the blocks and capacitors times must also be treated as inputs of the system. The Transformation Directive TD5 presents how the state succession operation inputs may be identified based on the relay-based RIS formalisation and the system relay diagram.

Transformation Directive TD5 (Inputs Identification): The inputs of the system state succession are the levers, buttons and relays presented in the Structural Definition SD1 and differentiated in the Structural Definition SD2. However, only the relays whose coils are not represented in the incidence function of the system graph representation are considered as inputs. Furthermore, for each block and capacitor, it must be defined one and two inputs, respectively, representing the passing of time required for these components behaviour, as presented in the Behavioural Definitions BD16 and BD20.

The relays that are considered as inputs are the ones whose contacts are part of the system circuit although their coils are represented externally. In this context, the relay is controlled by the environment, but it has influence over the system state. Figure 4.3 presents an example of the use of an external relay in the ITCS case study. In this figure, it is possible to visualise how an external relay, controlled by the environment, has influence over the system electrical circuit through a contact.

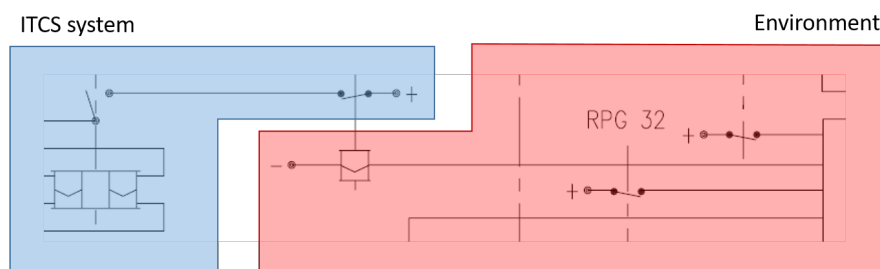


Figure 4.3 – Part of the ITCS example showing the separation between the ITCS system and its environment, which are connected by an external relay.

The identification of the inputs is visually simple when analysing the relay diagram, since buttons and relays have unique appearances and external relays are not represented inside the system electrical circuit. Nonetheless, the analysis of the graph incidence function and the components differentiation definition presented in the relay-based RIS formalisation also offers a strong option for the system inputs identification. Regarding the timed components, one may not identify the need of defining the passing of time in the formal specification by analysing the relay diagrams. The knowledge offered by the RIS formalisation is essential in order to specify the time as an input of the system.

In the ITCS case study, the formal specification of the state evolution begins with:

```
update_poste_A(L_C_CSS, INT_AC_V2, EPA_C_CSS, EIT_C_912, KAG_a_G,
              RPD_FA_C_911, L_ITCS, KAU_V2, KSS_E_V2, EPA_C_911,
              TA_SS_E_V2_Time),
```

where the list of the system inputs is defined inside the parenthesis after the operation name "update_poste_A". In this example, some of the inputs depicted in the relay diagram are not specified, like the buttons and some of the external relays, for instance. The buttons in this system are responsible for allowing the possibility of activating the ITCS system. As the formal specification of this system is focused on the verification of the ITCS, the buttons states can be ignored. In this context, we consider that an ITCS can always be used, which is the expected situation. Some of the external relays are not represented as their existence depend on the system environment. One example is the relay EPA_C_912, which is related to the existence of another turnout between the two control areas. In fact, in determined situations, these relays may not exist, so we are considering here only the most simple example of an ITCS system. The input for the time of the block TA_SS_E_V2 is represented by the input TA_SS_E_V2_Time.

As well as the B variables, the inputs must also be typed. Although these components are not part of the system state space, their values are important in order to define the system state evolution. In B, all the inputs are typed inside the operation precondition after the reserved word PRE. Similarly to variables monostable relays states are represented as Boolean values, as presented in the Behavioural Definition BD8. In the same manner, buttons states and the passing of time for blocks and capacitors are also represented as Boolean values as defined in the Behavioural Definitions BD3, BD16 and BD20, respectively.

Nevertheless, levers and bistable relays are special cases that require a little more attention. The relay-based RIS formalisation proposes a general representation of these components states as *right* and *left* or *config_a* and *config_b* so it can be used in any example. As this formalisation is based on a structural model, the meaning of these states is given by the structure itself. The formal specification proposed in this chapter, however, abstracts the system structure and focuses on the specification of specific systems instead of providing a general system behaviour. In this context, instead of determining general names for the levers and bistable states, one may use the ones represented inside the diagram. As a result, this modelling decision provides a stronger link between the initial structural model and the final behavioural specification. Regarding the bistable relays, an example of how these components states are presented in the relay diagram is depicted in Figure 4.2. Figure 4.4 illustrates how the same states are represented for the lever L_ITCS. In this case, when the lever is in the ES state, it allows the current to flow in the upper contact. Otherwise, in the DV state, the current flows only in the lower contact. By modelling this component type in this manner, it is also possible to abstract the relation between the levers and their contacts, as well as between bistable relays and their coils which is presented in the Structural Definitions SD3 and SD6, respectively.

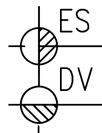


Figure 4.4 – Lever L_ITCS of the ITCS case study representing the states that this component may assume.

The mapping between the representation of the inputs values in the RIS formalisation and in the B formal specification is presented in Table 4.3. The Transformation Directive TD6 details

how the system inputs are typed inside the relay-based RIS formal specification.

Transformation Directive TD6 (Inputs Typing): The type of each input is defined as presented in the RIS formalisation behavioural definitions. For monostable relays and buttons, their values are represented by Boolean values, as described in the Behavioural Definitions BD8 and BD3, respectively. Similarly, the blocks and capacitors passing of time inputs are also defined as Boolean values, as presented in the Behavioural Definitions BD16 and BD20. The type of bistable relays and lever, however, is defined accordingly to the Behavioural Definitions BD9 and BD4, but their states are replaced by the positions established in the relay diagram.

Table 4.3 – Inputs values representation mapping.

Component	RIS formalisation	Formal specification
Monostable relays	Boolean	Boolean
Bistable relay	<i>right</i> or <i>left</i>	Specific representation inside the relay diagram
Buttons	Boolean	Boolean
Levers	<i>config_a</i> or <i>config_b</i>	Specific representation inside the relay diagram
Time	Boolean	Boolean

In the ITCS case study, for instance, the types of the inputs are defined inside the operation precondition as:

```
PRE L_C_CSS : O_OR_F & INT_AC_V2 : BOOL & EPA_C_CSS : BOOL &
    EIT_C_912 : BOOL & KAG_a_G : BOOL & RPD_FA_C_911 : BOOL &
    L_ITCS : DV_OR_ES & KAU_V2 : BOOL & KSS_E_V2 : BOOL &
    EPA_C_911 : BOOL & TA_SS_E_V2_Time : BOOL
```

An important adaptation made in this specification that may be observed is the type of the bistable relay EIT_C_912, which is defined as a Boolean input. This is a modelling decision in order to avoid using the values *right* and *left* as the possible states of this component is not presented in the relay diagram. In this case, the TRUE state represents that the contact is closed, i.e., in the *left* state.

Once defined the inputs and their types, one must describe the logic for the state evolution. As the state succession is not the objective of the relay-based RIS formalisation, the notation used for its specification is entirely defined by the B-method notation. With the aim of evolving the state of all the variables at once, it is possible to use a native B-method expression whose structure can be denoted as:

```
<<variables>>:(<<variables typing>> & <<logic>>).
```

By using this expression, it is possible to change the value of all the variables (<<variables>>) by reaffirming their types (<<variables typing>>) and describing the logic for the state evolution (<<logic>>). This system state evolution notation and its use is detailed in the Transformation Directive TD7.

Transformation Directive TD7 (State Evolution Notation): The system state evolution is specified inside a unique operation responsible for defining the state of all the variable at once according to the inputs given. With this purpose, the notation <<variables>>:(<<variables

typing>> & <<logic>>) is used, where <<variables>> contains the list of variables defined in the Transformation Directive TD1 and <<variables typing>> contains the variable typing described in the Transformation Directive TD2. The logic for the variables state succession is then defined inside the <<logic>> part of this notation.

In order to determine the state of a component according to the given inputs, one must specify the precondition for this component to reach its states. This logic has already been presented in the relay-based RIS formalisation as it denotes the relation between the components states. Regarding the outputs, for instance, their states are defined in Boolean values based on their activation, as presented in the Behavioural Definition BD14. The precondition for the activation of these components is presented in the Behavioural Definition BD15, which determines that an output is activated as soon as it is electrified.

In this context, the components electrification conditions presented in the Behavioural Definitions BD7, BD19 and BD23 describe in a general manner how monostable relays, bistable relays coils and outputs may be electrified based on the formalised structural model. This condition is complex as it uses first order logic in order to define a general property. In this chapter, it is possible to use this definition as basis in order to specify each component state precondition in a propositional logic expression.

In this context, as the outputs are activated when they are electrified, one may consider that the precondition for this component to be activated (TRUE) is the existence of at least one path between two energy sources allowing the current to flow inside this component. The existence of each of these paths depend exclusively on the contacts (relays), buttons and levers contacts (levers) states, as detailed in the Behavioural Definitions BD5 and BD6. One may then conclude that the condition for the activation of an output is the closing of the contacts, buttons and lever contacts of at least one path that may electrify it. In certain cases, these conditions may also include the blocks and capacitors states, as these components may provide energy to the system in determined occasions. In this context, we define as "electrified path" every path that has current flowing through it. The specification of the output state succession is presented in the Transformation Directive TD8

Transformation Directive TD8 (Output State Succession): As determined in the Behavioural Definition BD15, the outputs are activated once electrified. So, the precondition for the activation of an output is the existence of a path that may activate it, which may be specified as the disjunction of the conditions for the existence of each electrified path. This is denoted in B in the format $output = bool(path1 \text{ or } path2 \text{ or } \dots \text{ or } pathx)$ for the condition for the existence of the electrified $path1$ to $pathx$ that may electrify the component $output$.

In the ITCS case study, for instance, the activation of the output $EF11$ depends exclusively on the existence of the path:

$$\langle P1, SS_E_V2_1, C_CSS_V2_1, EIT_C_CSS_1, INT_AC_V2_1, PG_911_1, EF11, N1 \rangle.$$

Based on the definition of the components real connections and the system disconnections presented in the Behavioural Definitions BD6 and BD5, respectively, One may conclude that this component is activated if:

- $SS_E_V2_1$ is in the *down* state,

- $C_CSS_V2_1$ is in the ES (*right*) position,
- $EIT_C_CSS_1$ is in the F (*left*) position and
- $INT_AC_V2_1$ and PG_911 are in the *up* state,

which is also the condition for the existence of the path. As it is possible to abstract the contacts states since they are part of the relay, one may affirm that the component EF11 is activated if:

- SS_E_V2 is deactivated,
- C_CSS_V2 is in the ES position,
- EIT_C_CSS is in the F position and
- INT_AC_V2 and PG_911 are activated.

This condition may be written in B-method using the *bool*(*<< condition >>*) notation, which returns a Boolean value based on the propositional expression specified inside the parenthesis:

$$EF11 = \text{bool}(SS_E_V2 = \text{FALSE} \ \& \ C_CSS_V2 = \text{POS_ES} \ \& \ EIT_C_CSS = \text{POS_F} \ \& \ INT_AC_V2 = \text{TRUE} \ \& \ PG_911 = \text{TRUE}) .$$

In the case where more than one path may activate an output, this condition is defined as the disjunction of the conditions for the existence of each electrified path. These conditions may be defined based on the relay diagram model, as the drawing makes it possible to visualise the possible paths that may electrify one component. Nonetheless, the relay-based RIS formalisation may be used as a way to provide a support for this transformation by giving a mathematical strong background that may enable an automatic (or even partially automated) transformation and verification process.

Regarding the monostable relays, their activation also depends exclusively on their electrification, as presented in the Behavioural Definition BD10. In this context, the monostable relays state succession is defined in the Transformation Directive TD9.

Transformation Directive TD9 (Monostable Relays State Succession): As determined in the Behavioural Definition BD10, the monostable relays are activated once electrified. So, the precondition for the activation of a monostable relay is the existence of a path that may activate it, which may be specified as the disjunction of the condition for the existence of each electrified path. This is denoted in B-method in the format $\text{monoRelay} = \text{bool}(\text{path1} \ \text{or} \ \text{path2} \ \text{or} \ \dots \ \text{or} \ \text{pathx})$ for the condition for the existence of the electrified *path1* to *pathx* that may electrify the component *monoRelay*.

This condition is valid to all monostable relays that are part of the specification variables, nonetheless it is possible to make use of the state succession notations of B-method in order to improve this specification in determined situations. One may, for instance, consider the existence of self-powered relays, which are the ones whose contacts are part of the path that may electrify it. One example of this type of relay is the component PG_911 of the ITCS case study, as depicted in Figure 4.5. Although the Transformation Directive TD9 is correct, this is a special situation since this component may never physically be responsible for activating itself. This is because its contact only closes when the component is activated. Moreover, this

contact may never cause the component deactivation as it only opens when the component is deactivated. The Transformation Directive TD9 is correct and works fine, but it can be adapted to self-powered relays so it can be more physically accurate.

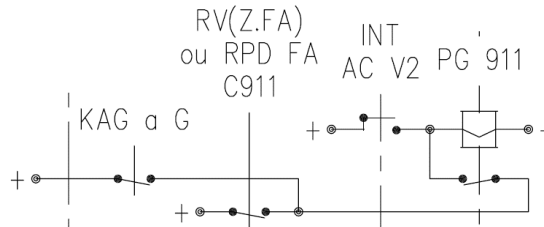


Figure 4.5 – Example of a self-powered relay in the ITCS case study.

In order to define the self-powered monostable relays activation precondition, one may use the B-method state succession notation "\$0". In this context, given a variable a , the notation $a\$0$ results in the value of the variable a in the previous system state. It is useful in this context in order to define that:

- If the self-powered monostable relay is deactivated, only the paths that do not contain its component contacts may activate it, and
- If the self-powered monostable relay is activated, the state of its contacts must not be taken into consideration for the relay deactivation condition.

This is formalised in the Transformation Directive TD10. Although this self-powered state precondition is more accurate with the real electrical circuit behaviour, the result of this new definition is as logically accurate as the expression presented in the Transformation Directive TD9.

Transformation Directive TD10 (Self-powered Monostable Relays State Succession): The precondition for the activation of self-powered monostable relays follows the condition presented in the Transformation Directive TD9, but it can be adapted in order to be more accurate with the real physical situation. In this case, the specification of the new precondition is conform to the notation:

$$\begin{aligned} & (mr\$0 = \text{FALSE}) \Rightarrow \text{bool}(\langle\langle\text{activation condition}\rangle\rangle) \\ & \& \\ & (mr\$0 = \text{TRUE}) \Rightarrow \text{bool}(\langle\langle\text{deactivation condition}\rangle\rangle). \end{aligned}$$

In this expression, the activation and deactivation conditions are the disjunction of the condition for the existence of each electrified path that activates and deactivate the component mr , respectively. In this context, these conditions must not depend on this same component state.

Regarding the component PG_911 , for instance, one may define the precondition for its activation using the Transformation Directive TD9 as:

$$\begin{aligned} PG_911 = & \text{bool}((RPD_FA_C_911 = \text{TRUE} \& PG_911 = \text{TRUE}) \text{ or} \\ & (KAG_a_G = \text{TRUE} \& PG_911 = \text{TRUE}) \text{ or} \\ & (INT_AC_V2 = \text{FALSE})). \end{aligned}$$

Although this logic is not wrong, it is physically illogical that this component state depends on its own state like this. So, the state succession of this component may be adapted based on the Transformation Directive TD10 as:

```
(PG_911$0 = FALSE => PG_911 = bool(INT_AC_V2 = FALSE)) &
(PG_911$0 = TRUE => PG_911 = bool(RPD_FA_C_911 = TRUE or
                                KAG_a_G = TRUE or
                                INT_AC_V2 = FALSE)).
```

In order to exemplify the notation presented in the Transformation Directive TD7, the specification of the state evolution of this component must be specified as:

```
...,PG_911,... :(... & PG_911 : BOOL & ...
&
(PG_911$0 = FALSE => PG_911 = bool(INT_AC_V2 = FALSE)) &
(PG_911$0 = TRUE => PG_911 = bool(RPD_FA_C_911 = TRUE or
                                KAG_a_G = TRUE or
                                INT_AC_V2 = FALSE)) &
...).
```

This B-method notation for acquiring a variable value in the previous state is also useful in order to define the bistable relays states. When both coils of a bistable relay are deactivated or activated, its state is maintained, as presented in the Behavioural Definition BD11. Thus the previous system situation is extremely important in order to define this component state evolution. The specification of the bistable relay state succession is defined in the Transformation Directive TD11.

Transformation Directive TD11 (Bistable Relay State Succession): According to the Behavioural Definition BD11, a bistable relay assumes the right state if only the right coil is activated and it assumes the left state if only the left coil is activated. Otherwise, the state of this component is maintained accordingly to the previous system state. As the bistable relay coils are activated once electrified, the definition of their states follow the same rules presented for outputs and monostable relays. In this context, one may define the precondition for the bistable relays states as:

```
(br$0 = right) =>
  (br = {TRUE |-> left, FALSE |-> right}
   bool(<<condition for the activation of the left coil while the
        right one is deactivated>>))
&
(br$0 = left) =>
  (br = {TRUE |-> right, FALSE |-> left}
   (bool(<<condition for the activation of the right coil while the
        left one is deactivated>>))).
```

This notation defines that, if the bistable relay is in the right state, it goes to the left state if only the coil in the left side is activated (the condition is true). Otherwise, this component state

remains the same. On the other hand, if the bistable relay is in the left state, it goes to the right state if only the coil in the right side is activated (the condition is true). In cases where both coils are activated or deactivated, the bistable relay maintains its previous state.

Nonetheless, instead of using the *right* and *left*, we propose the use of the states presented in the relay diagram, which makes the formal specification to be closer to the initial structural model, enhancing its comprehension. The relation between the coils and the relays presented in the Structural Definition SD6 is abstracted here by explicitly defining the relay states through its coils activation conditions. In the ITCS case study, for instance, the component *EIT_C_CSS* is one example of bistable relay that must be specified using this logic. The specification of the preconditions for the states of this component is:

```
(EIT_C_CSS$0 = POS_0 =>
  EIT_C_CSS = { TRUE |-> POS_F, FALSE |-> POS_0 }
  (bool(L_C_CSS = POS_F & EPA_C_CSS = TRUE)))
&
(EIT_C_CSS$0 = POS_F =>
  EIT_C_CSS = { TRUE |-> POS_0, FALSE |-> POS_F }
  C_CSS_V2 = POS_ES & EIT_C_912 = FALSE)).
```

Regarding the precondition for the blocks activation, they can be described similarly to the monostable relays activation. However instead of defining a path between the energy sources, the block activation may depend on the existence of a cycle that begins and finishes on the block independent connections, as presented in the Behavioural Definition BD18. Besides, the block may also be activated by a single connection between this component and an energy source through a block independent connection. Furthermore, as detailed in this same definition, it is important to consider the passing of time in order to determine the block state. In this B-method formal specification, the time is specified as an input of the operation and it is also represented by a Boolean value. The timed activation and timed deactivation blocks state succession are defined in the Transformation Directive TD12 and TD13, respectively.

Transformation Directive TD12 (Timed Activation Block State Succession): A timed activation block is activated if:

- there is a path that begins and finishes in this component independent connections and the time has passed, or
- there is a path between this component independent connection and an energy source and the time has passed.

The block activation condition may then be specified in B-method as: `block = bool((path1 or path2 or ... or pathx) & <<passing of time>>`, for the condition for the existence of the electrified paths `path1` to `pathx` that may electrify this component when the time for the block activation has passed. In this context `<<passing of time>>` must be replaced by a Boolean expression stating that the input for the passing of time for the block has the TRUE value.

Transformation Directive TD13 (Timed Deactivation Block State Succession): A timed deactivation block is activated if:

- there is a path that begins and finishes in this component independent connections,
- there is a path between this component independent connection and an energy source, or
- there is no path that electrify this component and the time has not passed.

The block activation condition may then be specified in B-method as: $\text{block} = \text{bool}(\text{path1 or path2 or } \dots \text{ or pathx})$, for the condition for the existence of the electrified paths path1 to pathx that may electrify this component; or $\text{block} = \text{bool}(\text{not}(\text{path1 or path2 or } \dots \text{ or pathx}) \ \& \ \ll\text{passing of time}\gg)$, stating that that there is not a path that electrifies this component and the time for the deactivation has not passed. In this context $\ll\text{passing of time}\gg$ must be replaced by a Boolean expression stating that the input for the passing of time for the block has the FALSE value.

So, one must use the activation condition defined in the Behavioural Definition BD18 as basis for the determination of the precondition of the block activation. In the ITCS, case study, for instance, considering the input TA_SS_E_V2_Time that specifies the passing of time for the block TA_SS_E_V2, the activation condition of this component may be defined in B-method as:

```
TA_SS_E_V2 = bool(TA_SS_E_V2_Time = TRUE & C_CSS_V2 = POS_ES &
  ((EIT_C_CSS = POS_0 & KSS_E_V2 = TRUE) or
   (SS_E_V2 = TRUE & EIT_C_CSS = POS_0) or
   (SS_E_V2 = TRUE & PG_911 = FALSE) or
   (SS_E_V2 = TRUE & INT_AC_V2 = FALSE))).
```

In this case it is important to note the existence of multiple cycles that may electrify this component, as presented in the relay diagram. Nonetheless, the relay SS_E_V2 is only activated when the block is also activated as it is connected to the block dependent connections. Accordingly to the Behavioural Definition BD19, the precondition for the activation of the component SS_E_V2 can be simply specified as:

```
SS_E_V2 = bool(TA_SS_E_V2 = TRUE).
```

It is important to notice that this relay contains a contact that is part of the cycle that may electrify the block TA_SS_E_V2. As a consequence, similarly to the relays that that are electrified by its own contacts, it is possible to conclude that the block cannot be directly activated by this component. Although the block activation logic is not incorrect, it can also be simplified by the use of the state succession notation as a way to be closer to the real situation:

```
(TA_SS_E_V2$0 = FALSE =>
  TA_SS_E_V2 = bool(TA_SS_E_V2_Time = TRUE & C_CSS_V2 = POS_ES &
    EIT_C_CSS = POS_0 & KSS_E_V2 = TRUE))
&
(TA_SS_E_V2$0 = TRUE =>
  TA_SS_E_V2 = bool(TA_SS_E_V2_Time = TRUE & C_CSS_V2 = POS_ES &
    (EIT_C_CSS = POS_0 or PG_911 = FALSE or INT_AC_V2 = FALSE))).
```

However, in this approach we chose not to enforce this adaptation by defining a transformation directive as it is not so obvious for some readers. Nonetheless, it is important to consider the possibility of making this logical simplification.

The last component whose specification is presented in this approach is the capacitor. Although this component is not used in the ITCS example, it is important to define how one may formally specify it based on the relay-based RIS formalisation. The differentiation between the capacitors given by the Structural Definition SD12 simplified this component state definition, allowing one to state that a capacitor is activated if:

- it is connected to the energy sources and the time for its activation has passed or
- it is no longer connected to the energy sources and the time for its deactivation has not passed.

These conditions may be formally specified in B-method as presented in the Transformation Directive TD14.

Transformation Directive TD14 (Capacitors State Succession): The precondition for the capacitor activation may be formalised in the format $\text{cap} = \text{bool}(\langle\langle\text{activation condition}\rangle\rangle)$, where cap is the capacitor variable and $\langle\langle\text{activation condition}\rangle\rangle$ consists on:

- the disjunction of the conditions for the existence of the electrified paths that contain this component, and the time for the capacitor activation has passed, which is denoted in the format:

$$((\text{path1 or path2 or } \dots \text{ or pathx}) \ \& \ \text{capActTime} = \text{TRUE});$$

- the negation of the disjunction of the conditions for the existence of the electrified paths that contain this component (the component is not electrified), and the time for the capacitor deactivation has not passed, which is denoted in the format:

$$(\text{not}(\text{path1 or path2 or } \dots \text{ or pathx}) \ \& \ \text{capDeactTime} = \text{TRUE}).$$

In this definition, capActTime and capDeactTime are the inputs representing the passing of time for the capacitor cap activation and deactivation, respectively.

Based on these transformation directives for the RIS formalisation logic adaptation in order to conform to the B-method Boolean logic, the operation for the ITCS state evolution, named as update_poste_A , can be specified as:

```

update_poste_A(L_C_CSS, INT_AC_V2, EPA_C_CSS, EIT_C_912, KAG_a_G,
  RPD_FA_C_911, L_ITCS, KAU_V2, KSS_E_V2, EPA_C_911, TA_SS_E_V2_Time) =
PRE L_C_CSS : 0_OR_F & INT_AC_V2 : BOOL & EPA_C_CSS : BOOL &
  EIT_C_912 : BOOL & KAG_a_G : BOOL & RPD_FA_C_911 : BOOL &
  L_ITCS : DV_OR_ES & KAU_V2 : BOOL & KSS_E_V2 : BOOL &
  EPA_C_911 : BOOL & TA_SS_E_V2_Time : BOOL
THEN KIT_C_CSS, KIT_C_911, EIT_C_CSS, PG_911, C_CSS_V2, EF11,
  TA_SS_E_V2, SS_E_V2:(

```

```

KIT_C_CSS : BOOL & KIT_C_911 : BOOL & EIT_C_CSS : O_OR_F &
PG_911 : BOOL & C_CSS_V2 : DV_OR_ES & EF11 : BOOL &
SS_E_V2 : BOOL & TA_SS_E_V2 : BOOL &

KIT_C_CSS = bool(SS_E_V2 = TRUE & EIT_C_CSS = POS_0 &
                INT_AC_V2 = TRUE & L_C_CSS = POS_0) &

KIT_C_911 = bool(KIT_C_CSS = TRUE & KAU_V2 = TRUE &
                C_CSS_V2 = POS_ES & PG_911 = TRUE) &

(EIT_C_CSS$0 = POS_0 =>
  EIT_C_CSS = {TRUE |-> POS_F, FALSE |-> POS_0}
              (bool(L_C_CSS = POS_F & EPA_C_CSS = TRUE))) &
(EIT_C_CSS$0 = POS_F =>
  EIT_C_CSS = {TRUE |-> POS_0, FALSE |-> POS_F}
              (bool(L_C_CSS = POS_0 & C_CSS_V2 = POS_ES &
                    EIT_C_912 = FALSE))) &

(PG_911$0 = FALSE => PG_911 = bool(INT_AC_V2 = FALSE)) &
(PG_911$0 = TRUE =>
  PG_911 = bool(RPD_FA_C_911 = TRUE or
                KAG_a_G = TRUE or INT_AC_V2 = FALSE)) &

(C_CSS_V2$0 = POS_ES =>
  C_CSS_V2 = {TRUE |-> POS_DV, FALSE |-> POS_ES}
            (bool(L_ITCS = POS_DV & EPA_C_CSS = TRUE &
                  EPA_C_911 = TRUE))) &
(C_CSS_V2$0 = POS_DV =>
  C_CSS_V2 = {TRUE |-> POS_ES, FALSE |-> POS_DV}
            (bool(L_ITCS = POS_ES & EPA_C_CSS = TRUE &
                  EPA_C_911 = TRUE))) &

EF11 = bool(SS_E_V2 = FALSE & C_CSS_V2 = POS_ES &
            EIT_C_CSS = POS_F & INT_AC_V2 = TRUE &
            PG_911 = TRUE) &

SS_E_V2 = bool(TA_SS_E_V2 = TRUE) &

(TA_SS_E_V2$0 = FALSE =>
  TA_SS_E_V2 = bool(TA_SS_E_V2_Time = TRUE & C_CSS_V2 = POS_ES &
                    EIT_C_CSS = POS_0 & SS_E_V2 = TRUE)) &
(TA_SS_E_V2$0 = TRUE =>
  TA_SS_E_V2 = bool(TA_SS_E_V2_Time = TRUE & C_CSS_V2 = POS_ES &

```

$$((\text{EIT_C_CSS} = \text{POS_0} \text{ or } \text{PG_911} = \text{FALSE} \text{ or } \text{INT_AC_V2} = \text{FALSE})))$$

The knowledge about the relay-based RIS behaviour formalised in Chapter 3 supports the formal specification of these systems by providing the relation between the components states. The specification of these systems in B-method allows the use of the supporting tools of this language in order to perform automatic analysis and verification as a way to guarantee the system safety. In the next section, it is provided a discussion about the formalisation support and the benefits of using the formal specification language.

4.3 Discussion

The relay-based RIS formalisation presented in Chapter 3 provides a set of structural and behavioural definitions that support the formal specification of these systems. The relay diagrams, which are used as basis for this formalisation, only describe the structure of the relay-based RIS by depicting the physical relation between the components. Then, we present in this thesis a formalisation of the system structure and use it as basis for the creation of a behavioural model. The formal specification based on these models is focused on describing the system behaviour as a way to be able to prove its safety. The relation between the structural/behavioural definitions and the formal specification is presented in Table 4.4, depicting how the information presented in the formalisation can support the formal specification presented in this chapter.

The identification of the machine variables is based on the components set and on the incidence function of the graph model of the system structure presented in the Structural Definition SD1. Then the differentiation between the components types presented in the Structural Definition SD2 is used in order to type the variables. In this case, the components states representation provided in the Behavioural Definitions BD8, BD9, BD14, BD17 and BD21 ground the components typing by detailing how the states of monostable relays, bistable relays, outputs, blocks and capacitors, respectively, may be described.

The Structural Definitions SD1 and SD2 are also used in order to identify and type the system inputs. In this context, the Behavioural Definitions BD8, BD9, BD3, BD4 provide information about the state representation of monostable relays, bistable relays, buttons and levers. Furthermore, the specification of inputs for the blocks and capacitors passing of time is grounded on the Behavioural Definitions BD16 and BD20, respectively.

The components state succession logic is specified accordingly to the components state conditions. These conditions are presented in the Behavioural Definitions BD10, BD11, BD15, BD18 and BD22, regarding monostable relays, bistable relays, outputs, blocks and capacitors, respectively. The outputs and relays states are grounded on the electrification conditions presented in the Behavioural Definitions BD7, BD19 and BD23. In this context, all the electrification conditions are based on the system real connections, and components disconnections, described in the Behavioural Definitions BD6 and BD5, respectively.

The rest of the structural and behavioural definitions present the relation between the components with their subcomponents, which is abstracted in this formalisation. Nonetheless, this information is useful in order to support the specification of the components state evolution. The relation between the bistable relays and its coils, for instance, is the key in order to determine

Table 4.4 – Behavioural and structural definitions that support the relay-based RIS formal specification.

B-clause	Definition	Given Support
VARIABLES	SD1	Lists the components that may become variables
INVARIANT	SD2	Differentiate these variables (components) into many different types
	BD8	Define the monostable relays type
	BD9	Define the type of the bistable relays
	BD14	Define the type of the outputs
	BD17	Define the type of blocks
	BD21	Define the type of capacitors
OPERATIONS (inputs definition)	SD1	Lists the components that may become inputs
	SD2	Differentiate these inputs (components) into many different types.
	BD8	Define the monostable relays type.
	BD9	Define the bistable relays type.
	BD3	Define the buttons type.
	BD4	Define the levers type.
	BD16	Define the blocks passing of time input.
BD20	Define the capacitors passing of time inputs.	
OPERATIONS (state evolution)	BD10	Determines the monostable relays states precondition.
	BD11	Determines the bistable relays states precondition.
	SD6	Abstracted relation between the bistable relays and their coils.
	BD15	Determines the outputs states precondition.
	BD18	Determines the blocks states precondition.
	BD22	Determines the capacitors states precondition.
	BD7, BD19 and BD23	Define the monostable relays, bistable relays coils and outputs electrification condition.
	SD4, SD7, BD1 and BD12	Abstracted relation between the monostable relays states and their contacts that basis the electrification conditions.
	SD5, SD8, BD2 and BD13	Abstracted relation between the bistable relays states and their contacts that basis the electrification conditions.
	SD3	Abstracted relation between the levers and their contacts that basis the electrification conditions.
	BD6 and BD5	System disconnections determining the system state that basis the electrification conditions.
	BD18, SD9 and SD10	Blocks electrification and activation condition based on the block types and connections.
	BD22, SD11, SD12, SD13	Capacitors electrification and activation condition based on the capacitors types, their relation with their plates and the plates potential charges.

these components states. The same applies to capacitors and their plates. Regarding blocks, the differentiation between its types is extremely important for the formal specification of these components behaviour. The relation between the levers and their contacts, on the other hand, are essential in order to understand and describe the components electrification condition.

The complete formal specification of the relay-based RIS in B-method allows the use of this language supporting tools in order to perform formal verifications. One example is the use of the ProB model checker, which exhaustively analyses if a system state may be inconsistent with the invariant defined. In the ITCS case study for instance, the verification of the system regarding the defined safety condition checked the existing 16 different states and analysed 32.769 different transitions between them in a little less than 3 seconds (2921ms in average). The verification was made by a 64bits Intel(R) Core(TM) i7-7600U 2.80GHz CPU with 16Gb RAM and running the Windows 10 operating system in its professional version.

A faster verification can be made with the use of Atelier B, which uses logical Proof Obligations (PO) in order to verify the consistency of the machine. In this context, using the logic foundations of the B-method, it is possible to guarantee that the state transitions specified in the machine operations never lead the machine to a state where the invariant is not true. Although Atelier B does not provide means to measure the time spent on verification, the use of logic for this purpose tends to have faster results when compared with model checking, since the former does not require an analysis of every possible system state.

In this work, the formal specification of the ITCS case study was verified with the use of ProB by model checking. Besides, we verified the same formal specification a second time with the Atelier B by theorem proving. Both tools were able to automatically prove the system without any human intervention. The result of this verification states that no error or inconsistencies have been found. Thus, one may conclude that the system will not lead to a dangerous state. The formal specification of this case study was produced together with specialists from the Railway and Formal Methods with the objective of creating a correct definition of the system.

It is important to note that the verification of the system using this formal specification considers all possible combination of inputs as a way to analyse all possible system states. In this context, the system verification is compositional at the diagram level, as it allows the complete verification of a diagram by considering all the possible combination of input values. Nonetheless, this approach has an important disadvantage: it may consider unrealistic states. As some of the inputs may be related in the environment, some inputs configurations may not exist in a real situation, which allows us to verify unpractical states. In fact, the verification we propose in this chapter is stronger than what it should be. Nonetheless, if the verification of a system finds unsafe states, it is necessary to analyse the environment and find if this state is indeed reachable or not.

A solution to this problem, as presented as a perspective in the conclusions of this work, is the use of conceptual models about the system environment as a way to enrich our specification. As our formalisation of the relay-based RIS is focused only on a relay diagram, the information about the system environment is lost. A conceptual model can be used in order to represent the knowledge about the system environment as a way to describe implicit relations between the components that are not presented in the relay diagram. Then, this information can be used as basis for specifying the state evolution operation precondition as a way to limit the combination of the inputs values and, by consequence, avoid unreachable states.

Regarding the traceability of problems, as the formal specification is supported by the behavioural model which is, in turn, supported by the structural formalisation, it is possible to analyse an error on the system behaviour and find possible structural causes. This is because the

behavioural model presents the relation between all the components states, thus, an inconsistent state can be analysed with the support of this model. Once the problematic component is found, it is possible to use the structural model as a way to analyse its connections and propose structural changes. In this context, the formalisation of the system presented in the Chapter 3 is the link between the B-method behavioural formal specification and the relay-diagrams structural models.

A limitation in this thesis is that the ITCS case study has no capacitors, thus it is not possible to evaluate the formal specification of this component. The formal specification of a new case study containing this component is in our near future agenda. This new case study also contains the use of alternated current, which may allow us to extend the transformation approach in order to contemplate the formal specification of more complex installations.

Part III

Conclusions

Conclusions and Perspectives

Conclusions

Railway Interlocking Systems are safety-critical systems and must be carefully developed in order to avoid the occurrence of hazardous situations. These systems can be implemented with many different technologies, like relay or computer-based. While the former has been historically used and safety tested for decades, the latter is an innovative technology that disposes of many benefits like a better extensibility and maintainability. The railway industry has interest on using computer-based systems, but it is imperative to maintain or even improve the safety level of the legacy systems. In this present thesis, we propose a solution for formally specifying the relay-based RIS behaviour as a way to prove their safety and produce computer-based RIS by refinement. By using this approach, it is possible not only to produce safety-proved systems, but also to evolve the existing legacy ones, maintaining their logic and safety level.

The literature regarding the formal specification of Railway Interlocking Systems contain many examples of how Formal Methods may be applied to certain contexts. Nonetheless, the presented approaches are generally focused on specific systems in a way that their solutions cannot be generalised. Each country and each railway company has specific safety and design rules, which makes it difficult to make a general formal specification approach. Regarding the relay-based RIS, for instance, there is not a unique method for modelling their structure in the form of relay diagrams, which results in models with different sets of components, relations and semantics. In order to cope with this contextual problem, we propose the use of basic mathematical foundations in order to model the relay diagrams structure and behaviour. By making a model in such abstraction level, it is possible to easily extend and adapt it to conform to different context and rules. Furthermore, this proposition is focused on the french context, which uses a larger set of different components compared to other systems, making it easier to be adapted. As this formalised model is grounded on mathematical foundations, safety proof can be performed as a way to guarantee that the system modelled in the relay diagram is safe.

This mathematical formalisation of the relay-based RIS has also two other benefits. The first one is the possibility of adapting it in order to specify these systems with the use of formal specification languages. By using Logic and Set Theory as basis for modelling the system, it is possible to use it as basis for formally specifying the RIS in any formal specification language that contains the same mathematical basis. These languages are generally supported by many tools that allow the specification, analysis, verification and refinement of the systems in a way that it may give a better automated support to the process of safety proving and implementing the relay-based RIS as computer-based systems. Furthermore, as each language is focused in a

different aspect of the system (like communication, synchronisation or concurrency, for instance), one may be able to specify these systems in a formal language that is the most appropriated according to his objectives.

The second extra benefit of using mathematical foundations in order to formalise such systems is the creation of a model that is understandable by all the experts involved. Relay diagrams are easily comprehended by railway experts, but it requires a higher effort from Formal Methods specialists. On the other hand, formal specification languages are the main tool for Formal Methods specialists, but it is hardly comprehended by the railway experts. So, as Set Theory and Logic are the foundations of all these domains, the formalisation of the relay-based RIS grounded on these mathematical foundations is a solution in order to create a model that can be understood by the experts of both domains.

The mathematical formalisation of the information contained inside a relay diagram represents the system structure based on a graph representation of the relations between the electrical components. Then, this structure is used as basis for the definition of a behavioural model that formalises the relation between the electrical components states during the system execution. This structural and behavioural formalisation can be used as basis for the the formal specification of these systems in B for instance. This formal specification language has the same mathematical foundations of the formalisation presented in this thesis and it has been successfully used in the railway industry as a way to produce safety-proved systems. So, this thesis also presents an approach for the transformation of the behavioural relay-based RIS formalisation logic to conform to the B language syntax and semantics as a way to formally specify these systems and take advantage of the benefits provided by this language. As a result, it is possible to use the B-method supporting tools in order to automatically prove the system safety and benefit from the formal system development approach provided by this language as a way to produce computer-based RIS by refinement.

This work is part of the LCHIP project, which objectives the transformation of the existing relay-based RIS into safety-proved computer-controlled systems that can be executed in micro controllers. As a result, it is possible to create systems that are more extendable and maintainable, moreover, it is possible to reduce the production costs and improve the safety level. The approach for the formal specification of these systems in the B-method is one important step for this transformation as it provides means for the refinement and implementation of the safety-proved specification.

Using the approaches presented in this thesis, it is possible to formalise the details of a case study provided by the French National Railway Company (SNCF), the Temporary Reversed Direction Installation (ITCS - Installations Temporaires de Contre Sens). Then, it was possible to formally specify and verify it regarding a safety property with the objective of avoiding a frontal collision between two trains. The methodologies presented in this work provide a complete basis for the analysis and specification of this system, giving a first detailed view of the relay-based RIS that can be used for many different purposes. The approaches for the formalisation and specification of the relay-based RIS detailed in this present thesis creates many different research opportunities in this field.

Perspectives

The work performed in this thesis creates many research opportunities that have the potential to enrich the literature with innovative ideas. By creating a general adaptable and extensible model, it is possible to use it as basis for the formal specification of systems in different formal languages and focused on different railway contexts. The refinement and implementation of these systems as computer-based ones can also be discussed in a future work. Furthermore, one may also objective the improvement of the approaches presented in this thesis as a way to consider contextual information and verify other important attributes of these systems, like availability or security, for instance. Some of these opportunities are described with some details in this section.

Refinement and Computer-based Implementation of Relay-based RIS

The formal specification of the relay-based RIS in B is the first step in order to implement these systems as safety-proved computer-based systems. This is because the B-method supports the use of a complete formal development methodology based on a well defined approach for the system refinement and implementation. Moreover, the Atelier B tool, which is one of the most important supporting tools of this language, possesses many features for the automatic refinement and code generation that are extremely useful for the system development process.

Nonetheless, the transformation of the system abstract machine into a more concrete one may be performed in many different manners, since there is not a unique way of performing the model transformation. In order to standardise the transformation of the relay-based RIS specification into computer-based systems, we aim to make an extension of the approach presented in this thesis as a way to comprise the whole formal software development process. In this extension, instead of producing only the abstract machine, we also objective the creation of the system implementation that is a proved refinement of this machine, which can shorten the development process.

Furthermore, by providing the complete software development methodology, it is possible to guarantee the well handling of the system inputs and outputs, being able to standardise the system interface as a way to assure that they can communicate with the physical components. In this context, one may guarantee that the system inputs and outputs are explicitly pointed so they can be correctly attached to the physical electronic components.

We believe that the approaches presented in this thesis can be automated by software, i.e., given the relay diagram, all the models can be automatically generated based on the graph representation of the diagram and the given mathematical foundations. The automatic generation of the formal specification and its refined implementation can be the answer for the railway industry to easily generate computer-based systems from the existing relay diagrams. We have presented some first ideas for the automatic transformation of the relay diagrams in [Almeida Pereira, Malki, et al. 2018]. In order to automate this process, we proposed a relay diagrams meta-model that supports the modelling of these systems in a XML format as a way to be processed by computers. Then, based on the definition of transformation directives grounded on the relay-based and the formal specification language meta-models, we believe that the transformation between the models can be automated. The extension of this approach for the

generation of the system implementation must still be analysed and studied in a future work.

Conceptual Modelling of the RIS Environment to Support the Formal Specification

By analysing the relay diagrams, it is clear that we need the knowledge of the people involved, but which was never clearly documented. Some of these concepts are important to describe the relay-based RIS behaviour. Beyond that, the formal specification of this information can improve our models as a way to be able to indeed guarantee the complete system safety. In the ITCS case study, for instance, there is a relation between the turnout and the signal that is not modelled in the relay diagram, but which is generally known by the specialists. This relation is extremely important in order to guarantee that the system does not cause a derailment.

The most important conceptual information about these systems are related to the environment of the relay diagrams that cannot be modelled inside these diagrams. Concepts like the train position, track extension or even implicit relation between the component states can be extremely important in order to provide a more complete formal specification. We have provided some initial studies about this in the work presented in [Almeida Pereira, Debbach, et al. 2019], where it describes how conceptual modelling may provide the description of information that is not presented in the relay diagram as a way to enhance the system formal specification and safety guarantee. Nonetheless, we believe that much more can be done concerning this subject.

In the future, we aim to use conceptual modelling to describe not only the implicit relation between the diagram components, but also to provide the relation between components from different diagrams, their position in the tracks, the trains position and behaviour and many other information that may give support to a better system formal specification. In this context, the verification may be able to make a complete simulation of the system in a conceptual environment in order to analyse its applicability and limitations. As the relay diagrams contain several undocumented information that are hard to foresee, we believe that conceptual modelling can be very enriching in this field, providing general models that can be used to fill all the existing gaps.

Relay-based RIS CSP specification

Communicating Sequential Processes (CSP) [Schneider 2000] is a formal approach focused on the specification, verification and analysis of real-time concurrent systems. By using CSP, it is possible to express the behaviour of each component as a different concurrent process and synchronise them when necessary. Besides, it is possible to define assertions that may be verified by the supporting tools, like the model checker FDR4 [Gibson-Robinson et al. 2014].

In this context, CSP may be used in order to specify relay-based RIS by defining a process for the behaviour of each component. Indeed, the relay-based RIS behaviour is formed by the concurrent behaviour of each component that constitutes the system, thus, it is possible to use the CSP approach and supporting tools in order to verify concurrency aspects of the relay-based RIS behaviour. In this case, it is possible to specify the behaviour of monostable relays and

contacts as:

$$\begin{aligned} \text{monoRelayBehaviour}(R) = \\ \text{relay}.R.\text{activated} \rightarrow \text{relay}.R.\text{deactivated} \rightarrow \text{monoRelayBehaviour}(R) \end{aligned}$$

$$\begin{aligned} \text{monoContactBehaviour}(C) = \\ \text{contact}.C.\text{up} \rightarrow \text{contact}.C.\text{down} \rightarrow \text{monoContactBehaviour}(C), \end{aligned}$$

where \rightarrow , in this context, represents the prefix notation, i.e., the succession of events inside a process. In this example, R and C are identifiers of the relays and contacts, respectively. The language also provide several other notations for specifying the synchronisation of these process that may be useful in order to define a more realistic model for describing the system execution.

Furthermore, as B-method has not a basis in order to deal with concurrency, it may not be able to verify many problems that may occur during the execution of relay-based RIS, like Deadlock, Livelock or Liveness problems. A deadlock may occur when the system cannot continue functioning because two or more processes are waiting for resources or for the end of each other executions. In this case, a deadlock verification may be extremely useful in order to guarantee that the system modelled in a relay diagram will not stop its execution because of deadlock problems.

A livelock may occur when the system keeps executing internal events indefinitely. This may useful in order to guarantee the absence of an infinite chain effect of the systems in a way that it is impossible to reach a stationary state. A common example occurs when two relays are responsible for the activation of each other. In this situation, a relay activates the other successively in an infinite chain effect that results on the system overheating. A livelock verification may be extremely useful in this type of situation.

Another common problem of relay-based RIS occurs when the extreme care regarding safety aspects leads the system to stop in an successive sequence of states that prevents a train to move indefinitely. Sometimes, this is a intended behaviour as a way to avoid the occurrence of hazardous situations. As an example, when the pedal that detects the train presence breaks, it keeps informing that there is a train in the track as a way to prevent other trains to enter in this zone, which avoids collision. However, in this scenario, the trains cannot move forward until the occurrence of an external intervention. The liveness verification may be used in this case as a way to predict the possibility of some processes to stuck in determined situations as a dysfunctional analysis verification. Furthermore, deadlock and livelock verifications may also be used as a way to verify the safety and the well functioning of the system when determined components break. Indeed, the availability of the system is one of the main desirable properties in a railway system as presented in the railway technical standards, like the ER50126 [CENELEC 2017a], for instance.

The complete specification of relay-based RIS in CSP and the benefits from the use of this approach may still be studied in a future work. As RIS are essentially concurrent systems, the use of CSP as an specification approach seems to be a promising solution in order to deal with many concurrency problems inherent to these systems. Despite the absence of a certification by the CSP supporting tools when compared with B, CSP has a complete and well founded set

of notations that may allow the verification of properties regarding the relay-based RIS safety that the B-method does not support. The creation of a methodology for the verification of these systems using CSP is in our future agenda. In this future work, we aim to ground the CSP RIS specification on the same foundations presented in this thesis. Then, it is possible to make a comparison between the B and CSP approaches.

Implementation of Safe and Secure RIS Based on Formal Methods and Software Obfuscation

Besides reliability, availability, maintainability and safety, the CENELEC standards are also concerned with the Security of the Railway Systems. The ER50126 [CENELEC 2017a] characterises security as "*the resilience of a railway system to vandalism, malevolence and intentionally harmful human behaviour*". As we are proposing in this work the transformation of the relay-based RIS into computer-based systems, it is important to consider the possibility of ill-intended attackers to tamper with the system, which may cause unexpected accidents or disrupting the traffic.

The ER50129 [CENELEC 2018] divides security according to two kinds of threats resulting from unauthorised access to the signalling equipment: Physical Security and IT-Security. The former is related to the protection of the system against direct physical access to the equipment from unauthorised people. In this context, the railway infrastructure managers must take precautions in order to hide the equipment in a way that it cannot be accessed. Regarding IT-Security, it is concerned with the possibility of remote attacks through logical access to the signalling system in a software level. Attacks of this kind have the potential to manipulate the signalling components and affect the functional safety. So, the railway infrastructure managers must take precautions in order to avoid that attackers can manipulate the systems.

As this thesis proposes the implementation of the Railway Interlocking Systems, it is important to consider the security in order to support the system safety. One known solution for improving the IT-Security is the use of software obfuscation, which has the objective of making the programs more difficult to be read and understood by human readers, which increases the cost for attackers. Although the deobfuscation of a program is costly, it is still feasible, so, the code obfuscation also proposes specific strategies for protecting the program from tapering even when attackers achieve an understanding of its semantics.

As a perspective from this thesis, we aim to provide a methodology for obfuscating the final safety-proved computer-controlled Railway Interlocking Systems with the objective of improving their security. We have presented an initial discussion about it in [Martinez et al. in press]. We believe that formal specification methodologies can take advantage of the software obfuscation methodologies as a way to produce safety and secure systems. In this context, we objective to make an analysis of the insertion of software obfuscation techniques in the B-method system refinement, providing a better formal software development methodology.

Résumé Étendu en Français

Problématique et Motivation de la recherche

Les systèmes ferroviaires sont des systèmes critiques de sécurité, de fait une défaillance peut avoir des conséquences inacceptables comme la perte de vies humaines et des dommages matériels et environnementaux importants. Ainsi, les méthodologies utilisées pour mettre en œuvre ces installations doivent être capables de prouver la sécurité du système en évitant l'apparition de situations dangereuses. Les Systèmes d'Enclenchement Ferroviaire (SEF) sont une partie importante de la signalisation ferroviaire, puisqu'ils ont en charge la détection de présence des trains et le contrôle des feux et des aiguillages afin d'éviter les collisions et les déraillements. Malgré l'existence de systèmes informatiques, la majorité de ces systèmes sont encore mis en service avec des technologies historiques basées sur des relais.

Bien que les technologies les plus récentes présentent des améliorations en ce qui concerne l'extensibilité et la maintenabilité du système, les SEF à base de relais sont utilisés depuis des décennies et par conséquent maîtrisés, ce qui leur confère un niveau de confiance élevé. Cependant, avant leur mise en œuvre, les SEF à base de relais ne sont représentés que structurellement sous la forme de schémas de circuits électriques (schémas de relais). Ainsi, l'analyse comportementale de ces systèmes est réalisée via une inspection manuelle qui est sujette aux erreurs, ce qui n'est pas pleinement satisfaisant dans un contexte de sécurité critique.

Dans les versions récentes des normes ferroviaires européennes [CENELEC 2011], il est fortement recommandé d'utiliser des méthodologies formelles pendant le développement des systèmes ferroviaires. La raison en est que les méthodes formelles reposent sur des bases mathématiques solides qui permettent non seulement de créer un modèle formel du système, mais aussi de le vérifier et d'en prouver la sécurité. En outre, certaines méthodes formelles, comme la méthode B, par exemple, propose une approche de développement formel complète qui permet la spécification, la vérification et la mise en œuvre du système par raffinement, en prouvant sa sécurité à chaque étape. Pour cette raison, les méthodes de spécification formelles sont, à notre avis, la clé pour prouver la sécurité des SEF à relais existants et les transformer en systèmes informatiques par un processus de développement formel.

De nombreux travaux dans la littérature ont proposé la spécification et l'implémentation formelles des SEF à base de relais. Cependant, ces travaux sont difficiles à généraliser, car ils se concentrent sur la spécification formelle de systèmes issus de contextes spécifiques dans des langages spécifiques pour la vérification de propriétés particulières. Bien que chaque système à base de relais ait une base électrique commune -relais, contacts, câbles, ...- chaque compagnie ferroviaire modélise les diagrammes à relais de manière différente, de sorte que la création d'une

approche unique de spécification formelle des SEF est un défi scientifique et méthodologique. En outre, chaque méthode formelle est plus ou moins adaptée à la vérification de certaines propriétés du système vérifié, ce qui ajoute une dimension supplémentaire à l'entreprise de modélisation et vérification formelle des SEF.

Cette thèse propose une modélisation mathématique de la structure et du comportement des Systèmes d'Enclenchement Ferroviaire à base de relais. L'utilisation d'outils mathématiques, au lieu d'un langage de spécification formel particulier, permet aux travaux d'être adaptables pour les spécifications formelles des SEF de différentes entreprises ferroviaires quelque soit le langage formel cible considéré pour la spécification et la vérification, lorsqu'il utilise les mêmes fondements mathématiques. De plus, les mathématiques constituent une base commune pour les experts du secteur ferroviaire et des méthodes formelles, de sorte que ces modèles peuvent être mieux compris par de nombreux experts impliqués dans le projet.

En illustrant comment le modèle mathématique d'un système d'enclenchement peut être utilisé comme base pour sa spécification formelle, cette thèse propose également un ensemble de directives pour la spécification formelle des SEF en langage B, langage associé à une méthode de conception logicielle qui a une histoire reconnue tant dans le domaine scientifique que ferroviaire. Dans ces travaux, une étude de cas est développée : l'Installation Temporaire de Contre-Sens (ITCS), sa sécurité est ensuite vérifiée à l'aide des outils de support de la méthode B.

Contexte de la recherche

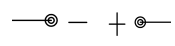

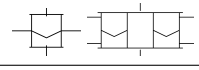
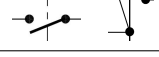
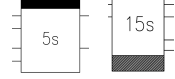
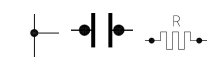
Cette thèse établit une synergie entre les domaines ferroviaire et les méthodes formelles, ce qui demande une attention particulière afin d'intégrer les connaissances, les expertises et les outils des deux domaines. Les systèmes à base de relais ont leur propre logique et leur propre fonctionnement. Par conséquent, le processus de spécification formelle de cette culture technologique particulière est une tâche délicate. Une analyse minutieuse de ces systèmes est donc impérative afin de soutenir leur spécification formelle. En outre, afin de modéliser ces systèmes uniquement sur la base de notations mathématiques, ce travail nécessite une connaissance solide de la logique, de la théorie des ensembles et des relations, qui sont les fondements les plus importants de notre méthodologie. Ces fondements mathématiques sont également utilisés par la méthode B comme moyen de soutenir la spécification formelle et la vérification du système.

Systèmes d'Enclenchement Ferroviaire à Base de Relais

Les Systèmes d'Enclenchement Ferroviaire sont la partie des systèmes de contrôle ferroviaire chargée de détecter la position des trains et de contrôler les aiguillages et les feux afin de garantir la sécurité du système. La logique de commande de ces composants électriques de voie peut être mise en œuvre grâce à l'utilisation de nombreuses technologies, comme les relais ou l'informatique.

Les SEF à relais sont la mise en œuvre de la logique d'enclenchement sous forme de circuits électriques. De nombreux composants électriques différents peuvent être utilisés dans ces systèmes, comme les sources d'énergie, les leviers, les boutons, les relais, les contacts, les blocs et les condensateurs, comme présenté dans le tableau 4.5.

Table 4.5 – Représentation des composants électriques à l'intérieur des diagrammes de relais.

	Sources d'énergie.
	Un levier et un bouton, respectivement.
	Relais monostable et bistable, respectivement.
	Un contact monostable et un contact bistable, respectivement.
	Blocs pour l'activation et la désactivation temporisées, respectivement.
	Une jonction, un condensateur et une résistance, respectivement.

Les sources d'énergie permettent d'électrifier les composants du système. Néanmoins, le flux de courant des fils peut être contrôlé par d'autres composants comme les leviers et les boutons. Ces composants font partie des entrées du système, permettant à l'environnement de modifier l'état de celui-ci. Un autre composant central peut contrôler le flux de courant au sein du système : le relais. Ce composant est divisé en deux types : monostable et bistable qui activent tous les deux des contacts agissant sur le passage du courant. Un relais monostable contient une bobine électromagnétique qui, une fois électrifiée, déplace les contacts mobiles dans une direction, le mouvement inverse étant le fait de la gravité. Un relais bistable, a contrario, contient deux bobines dont l'électrification permet un déplacement vers deux position stables -gauche et droite- pour les contacts.

Des blocs et des condensateurs sont utilisés dans ces systèmes afin de retarder l'activation ou la désactivation d'autres composants et introduisent l'aspect temporel. Les blocs d'activation temporisée sont représentés dans les schémas de relais par un trait plus épais en haut, tandis que les blocs de désactivation temporisée sont représentés par un trait plus épais en bas. Un condensateur peut être chargé lorsqu'il est connecté aux sources d'énergie. Une fois qu'un condensateur est chargé, il peut maintenir électrifiés les composants qui lui sont connectés jusqu'à ce que ce composant soit déchargé. Les condensateurs sont généralement utilisés avec des résistances comme moyen de contrôler le temps de charge ou de décharge de ce composant.

Tous ces composants peuvent être utilisés dans un schéma à relais afin de décrire une logique d'enclenchement. Dans la figure 4.6, l'étude de cas ITCS est présentée. Dans la situation normale, les deux voies sont dédiées à des directions opposées, les trains peuvent circuler librement entre les zones de contrôle A et C. Cependant, en raison d'un problème sur l'une des voies, on peut être obligé de condamner une voie. Il faut dans ce cas autoriser et permettre la circulation dans les deux sens sur la voie restante en sécurité. Par conséquent, le train qui vient de la zone de contrôle A doit passer par la direction opposée afin de continuer son chemin, ce qui peut provoquer une collision avec un train qui vient de la zone de contrôle C. Pour résoudre ce problème, il faut utiliser un système d'enclenchement capable de détecter la présence des trains et de contrôler les signaux en toute sécurité. Le schéma présenté dans la figure 4.6 représente le système utilisé dans la zone de contrôle A, où le composant *KIT_C_911* est responsable du contrôle du feu dans

cette zone et le composant *EF11* est responsable d'envoyer une information à la zone de contrôle C sur la disponibilité de la voie. Dans ce contexte, une condition de sécurité que ce système doit remplir est que ces deux composants ne soient jamais électrififiés en même temps, de sorte que les signaux dans les deux zones ne soient jamais ouverts -feu vert- simultanément.

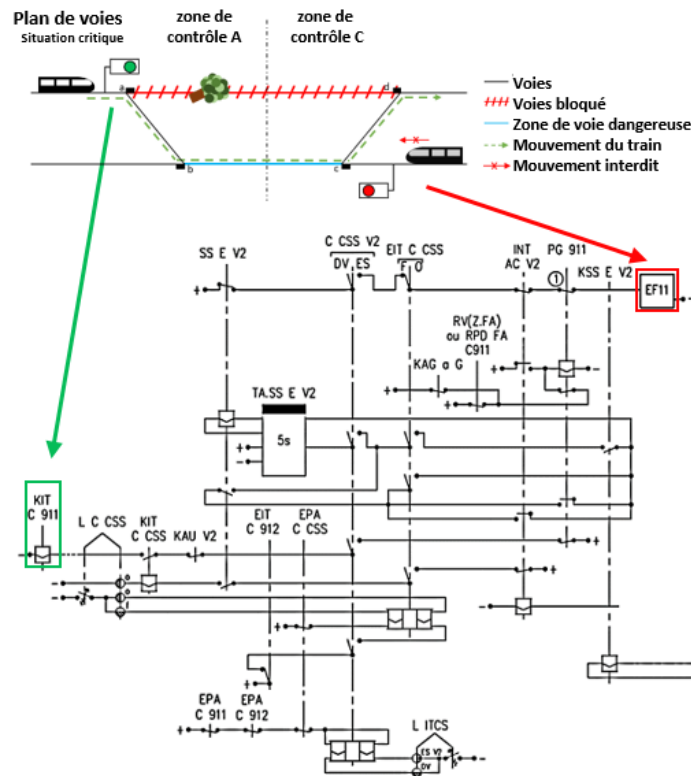


Figure 4.6 – Étude de cas de l'ITCS.

Un système d'enclenchement peut être composé de plusieurs schémas comme celui-ci, ce qui rend sa vérification difficile, longue et sujette aux erreurs. Afin de garantir la sécurité de ces systèmes, des méthodes formelles peuvent être utilisées, ce qui est fortement recommandé par les normes ferroviaires telles que la norme ER50128 [CENELEC 2011] du CENELEC (Comité Européen de Normalisation en Électronique et en Électrotechnique).

Fondements Mathématiques et Spécification Formelle

Les langages de spécification formelle comme B, Z ou les Réseaux de Petri sont basés sur des fondements mathématiques comme la logique et la théorie des ensembles. Ces fondements sont extrêmement importants pour fournir des moyens de prouver les propriétés du système spécifié. La Logique Propositionnelle et la Logique du Premier Ordre sont parmi les bases les plus importantes de l'analyse du système. En ce qui concerne la théorie des ensembles, elle peut être utilisée pour la définition d'ensembles, de relations, de fonctions et de séquences, ce qui apporte un soutien syntaxique et sémantique à la spécification formelle du système.

La logique propositionnelle permet la définition d'expressions avec des valeurs booléennes basées sur les notations logiques pour la négation des formules (\neg) ainsi que la conjonction (\wedge), la

disjonction (\vee), l'implication (\rightarrow) et la bi-implication (\leftrightarrow) entre les formules. Dans ce travail, la logique propositionnelle est utilisée afin de décrire les propriétés logiques garantissant la sécurité des SEF. Cependant, bien que la logique propositionnelle permette d'exprimer les relations entre les propositions individuelles, elle est insuffisante pour décrire des conditions plus générales. De ce fait, la logique du premier ordre l'étend en fournissant des outils afin de faire des hypothèses générales par l'ajout d'opérateurs relationnels et quantifiés. Alors que les opérateurs relationnels permettent la description de propriétés pour des variables déterminées, les opérateurs quantifiés utilisent les quantificateurs universels et existentiels (\forall et \exists , respectivement) afin de préciser la portée de ces dernières.

La théorie des ensembles, quant à elle, permet la spécification du système basée sur la définition des ensembles et leurs relations. La notion de relation dans ce contexte provient de la notation du produit cartésien (\times), où une relation entre deux ensembles A et B est définie comme l'ensemble de tous les sous-ensembles du produit cartésien compris entre A et B tel que $A \leftrightarrow B = \mathbb{P}(A \times B)$. Cette relation est une fonction si aucun élément de A n'est lié à plus d'un élément de B . Sur la base de ces définitions, une séquence d'éléments d'un ensemble C est définie comme une fonction de l'ensemble des nombres naturels positifs à C .

Le langage de spécification formelle B utilise ces bases mathématiques. La méthode B est une méthode formelle qui a été appliquée avec succès dans de nombreux projets industriels ferroviaires pour la spécification, la vérification, la preuve de sécurité et l'implémentation de systèmes ferroviaires. Cette méthode est soutenue par un certain nombre d'outils qui automatisent la vérification, la preuve de la sécurité et même la mise en œuvre des systèmes.

Dans la méthode B, une spécification est divisée en machines, qui sont à leur tour divisées en clauses. Chaque clause est responsable d'une information différente sur le système, comme, par exemple, les variables (VARIABLES), leurs valeurs initiales (INITIALISATION) et l'évolution de l'état du système (OPERATIONS). Dans [J.-R. Abrial 2005], il est présenté la syntaxe et la sémantique de B ainsi que des détails sur le processus de développement formel de la méthode B, y compris son raffinement et son processus de mise en œuvre avec des vérifications à chaque étape.

Analyse et Formalisation des Systèmes d'Enclenchement Ferroviaire à Base de Relais

La spécification formelle directe des Systèmes d'Enclenchement Ferroviaire à base de relais dans un langage de spécification formelle présente certains inconvénients :

- les experts ferroviaires n'ont généralement que peu d'expérience avec ces langages de spécification, ce qui nécessite une certaine formation afin que ces experts puissent travailler avec des experts en méthodes formelles dans le cadre d'un même projet ;
- le niveau d'abstraction plus élevé de ces langages rend difficile l'adaptation des approches de modélisation à différents contextes ferroviaires, ce qui résulte en la création de méthodes de spécification formelles pour les systèmes de compagnies spécifiques ;
- chaque langage est axé sur la vérification de différents aspects des systèmes (comme la

concurrence, la communication, l'évolution des états ou le traitement du temps), ce qui limite le pouvoir de vérification de ces méthodologies.

Afin d'essayer de résoudre ces problèmes, au lieu de spécifier directement et formellement les SEF dans un langage de spécification formel, cette thèse propose leur modélisation en utilisant les bases mathématiques. Ainsi, nous pouvons obtenir des modèles qui peuvent être compris par de nombreux experts impliqués en raison du contexte mathématique commun aux domaines ferroviaire et des méthodes formelles. En outre, en utilisant les mathématiques comme base pour la modélisation de ces systèmes, nous pouvons créer un modèle qui peut être plus facile à adapter à différents contextes et qui peut être utilisé comme base pour la spécification formelle et la mise en œuvre des SEF dans des langages qui soutiennent les mêmes fondements mathématiques.

Afin de modéliser ces systèmes en utilisant les notations mathématiques, nous devons tout d'abord séparer la structure et la logique du système. En effet, la structure du système est modélisée explicitement par les diagrammes de relais alors que le comportement n'est décrit qu'informellement en fonction du comportement spécifique de chaque composant. En ce qui concerne la structure du système, l'information la plus élémentaire présentée dans les diagrammes est la connexion électrique entre les composants par l'intermédiaire de fils. Ainsi, afin de modéliser la structure de ces systèmes, nous proposons dans cette thèse un modèle mathématique d'un graphe, où les nœuds d'un ensemble de composants (*Components*) sont connectés par des arêtes à partir d'un ensemble de fils (*Wires*) selon la fonction d'incidence : $Incidence_{\rightarrow} \in (Wires \rightarrow (Components \times Components))$. Dans ce contexte, afin de différencier les types de composants, nous proposons la création d'ensembles de sous-composants, comme *Levers*, *Buttons*, *PosSources* et *MonostableRelays* pour les leviers, boutons, sources d'énergie positive et relais monostables, respectivement.

En se basant sur la structure du système, il est possible de modéliser le comportement du système en fonction du comportement de chaque composant et de la relation entre leurs états. À titre d'exemple, les états des contacts monostables peuvent être définis en fonction des états que ces composants peuvent prendre : soit *up* soit *down*. Ainsi, on peut représenter les états de ces composants par une fonction allant de l'ensemble des contacts monostables à un ensemble contenant leurs états possibles : $MonoContactsSt_{\rightarrow} \in (MonostableContacts \rightarrow \{up, down\})$. Cependant, ces composants sont contrôlés par des relais monostables, dont les états (activé/désactivé) sont responsables du contrôle des positions des contacts monostables. Ainsi, on peut utiliser la logique pour déterminer que, lorsque le relais monostable est activé (représenté par la valeur booléenne *true*), les contacts correspondants prennent l'état *up* ; d'autre part, lorsque le relais monostable est désactivé (représenté par la valeur booléenne *false*), les contacts correspondants prennent l'état *down* :

$$\forall mc.(mc \in MonostableContacts \Rightarrow \\ ((MonoRelaysSt_{\rightarrow}(MonoRelayContactsRel_{\rightarrow}(mc)) \Rightarrow MonoContactsSt_{\rightarrow}(mc) = up) \wedge \\ (\neg MonoRelaysSt_{\rightarrow}(MonoRelayContactsRel_{\rightarrow}(mc)) \Rightarrow MonoContactsSt_{\rightarrow}(mc) = down)))$$

Dans ce contexte, $MonoRelaysSt_{\rightarrow}$ est la représentation des états des relais monostables comme la fonction : $MonoRelaysSt_{\rightarrow} \in (MonostableRelays \rightarrow \mathbb{B})$, où \mathbb{B} est l'ensemble booléen.

Les états des boutons et des leviers seront représentés par des notations similaires. En

fonction des états des contacts, des boutons et des leviers, il est possible de définir les conditions d'électrification des composants. En utilisant la représentation structurelle du graphe et les états des composants qui peuvent bloquer le flux de courant électrique, l'électrification et l'activation des composants peuvent également être définies sur la base d'expressions mathématiques. Dans ce cas, un composant est électrifié s'il existe un chemin dans la structure du graphe d'une source d'énergie positive à une source négative qui contient ce composant, en considérant les états des composants qui peuvent bloquer les connexions. L'existence de ce chemin indique qu'un courant électrique le traverse, il est donc électrifié.

En ce qui concerne les composants temporisés, l'électrification et l'activation de ces composants peuvent également être représentées sur la base des expressions mathématiques. Cependant, ces composants peuvent prendre un certain temps pour s'activer ou se désactiver (charge ou décharge). Dans ce contexte, nous considérons le temps comme une entrée de valeur booléenne du système indiquant si le temps d'activation/désactivation requis s'est écoulé pour ces composants. Ceci est dû au fait que ce modèle mathématique ne prend pas en compte les états transitoires, ainsi, seules les conditions préalables à l'activation/désactivation de ces composants sont importantes pour la définition de leur état.

Sur la base de ces modèles structurels et comportementaux généraux, nous sommes en mesure de décrire la structure et le comportement d'un système. En ce qui concerne la structure du système, il faut transcrire les informations contenues dans le diagramme de relais dans les ensembles et les relations du modèle structurel afin de représenter mathématiquement la structure du système. En ce qui concerne le comportement du système, le modèle comportemental présente déjà les états possibles des composants et leurs relations de manière à ce qu'aucune autre information ne soit nécessaire.

En utilisant la logique, on peut définir des propriétés de sécurité pour le système modélisé. En ce qui concerne l'exemple ITCS, une propriété de sécurité possible est que $\neg(\text{electrified}(\text{KIT_C_911}) \wedge \text{electrified}(\text{EF11}))$, ce qui impose que les composants *KIT_C_911* et *EF11* ne sont jamais électrifiés en même temps. Bien que les modèles puissent être utilisés pour la vérification formelle, leur objectif le plus important est de fournir une compréhension complète de la structure et du comportement du système, qui peut être utilisée comme base pour sa spécification formelle dans de nombreux langages de spécification formelle différents. En outre, en modifiant les définitions des ensembles et des fonctions, ces modèles peuvent être facilement adaptés à différents contextes ferroviaires présentant des caractéristiques différentes.

Afin d'automatiser la vérification de ces systèmes, une spécification formelle peut être nécessaire. B est un exemple de langage de spécification formelle qui peut être utilisé.

Spécification Formelle des Systèmes d'Enclenchement Ferroviaire en B

Une fois que le système est modélisé avec notre approche mathématique, nous pouvons utiliser ce modèle afin de le spécifier formellement. Afin de transformer le modèle en une spécification formelle en B, il est nécessaire de définir des directives de transformation qui décrivent comment chacune des clauses de la machine B peut être spécifiée à l'aide des informations modélisées. Dans ce contexte, il est important de prendre en compte que le modèle mathématique n'inclut

pas de logique de transition d'état, en conséquent certaines informations requises, comme l'état initial, par la méthode B manquant. En revanche, ces informations sont présentes dans le diagramme de relais et sont utiles pour compléter notre spécification formelle telle que présentée plus loin dans cette section.

La première clause de la méthode B qui doit être spécifiée est la clause `VARIABLES`, qui contient les variables système dont les valeurs possibles définissent l'espace d'état du système. Comme dans les SEF, cet espace d'état est défini par l'état possible de chaque composant électrique, on peut spécifier chaque composant électrique comme une variable. Néanmoins, il est important de considérer que les entrées du système ont une place particulière dans la spécification formelle de la clause `OPERATIONS`, de sorte qu'elles ne soient pas spécifiées comme des variables. En outre, il faut également considérer que la réduction du nombre de variables peut être utile pour le processus de vérification, en réduisant l'espace d'état du système et, par conséquent, le temps de vérification. C'est pourquoi nous avons décidé de ne pas diviser les composants en sous-composants dans la spécification formelle. Par exemple, les relais ne sont pas divisés en bobines et contacts.

La clause `INVARIANT` est utilisée afin de typer les composants et de décrire les propriétés qui doivent être satisfaites par le système. Dans ce contexte, les propriétés peuvent être utilisées pour décrire les conditions de sécurité, de sorte que nous pouvons vérifier la sécurité du système en analysant si l'invariant est respecté ou non. En ce qui concerne le typage des variables, nous avons décidé de définir l'électrification/activation des composants par des valeurs booléennes, soit `true` les états électrifés ou activés et `false` les états non électrifés/désactivés. Les valeurs booléennes peuvent également être utilisées pour déterminer si les boutons ferment les contacts (`true`) ou les ouvrent (`false`). Comme certains composants ont des états spéciaux, comme les leviers et les relais bistables, les états possibles de ces composants peuvent être définis dans la clause `SETS` en fonction des informations présentées dans le schéma des relais.

En ce qui concerne la clause `INITIALISATION`, nous ne pouvons pas la spécifier en nous basant sur les modèles mathématiques. En effet, ces modèles n'utilisent pas de logique de transition d'état et ne présentent donc pas d'état initial du système. Néanmoins, cette information peut être obtenue sur la base du dessin du diagramme de relais, qui représente un état fonctionnel du système.

En ce qui concerne la transition d'état du système, elle est spécifiée dans la clause `OPERATIONS`. Dans ce travail, nous n'utilisons qu'une seule opération pour décrire la transition complète de l'état du système puisque nous ne considérons pas les états transitoires. Ainsi, en utilisant une seule opération, nous sommes en mesure de considérer toutes les entrées du système en une seule fois et de générer l'état du système en fonction de ces entrées. La logique de l'évolution de l'état du système est alors définie sur la base de la relation entre les états des composants tels que définis dans le modèle mathématique comportemental.

La vérification d'un système spécifié à l'aide de notre méthodologie est basée sur l'analyse de tous les états possibles du système. Ainsi, un vérificateur de modèle, celui associé à la plateforme ProB, est utilisé afin de vérifier que toutes les combinaisons possibles de valeurs d'entrée conduiront le système à un état qui ne casse pas l'invariant du système. Dans ce travail, nous avons modélisé l'étude de cas de l'ITCS et l'avons spécifiée à l'aide de B. La vérification a conclu que les composants `KIT_C_911` et `EF11` ne sont jamais activés en même temps, ce qui est spécifié

comme une condition de sécurité dans l'invariant (`not (KIT_C_911 = TRUE & EF11=true)`).

Conclusions

Ce travail présente une formalisation de la structure et du comportement des diagrammes de relais en utilisant des notations mathématiques. Cette formalisation peut être utilisée pour l'analyse du système, bien que son objectif principal soit d'être une référence pour la spécification formelle du système, puisqu'elle décrit la relation structurelle et comportementale entre les composants du système. Ce travail fait partie d'un projet qui envisage la transformation des SEF existants basés sur des relais en des SEF informatisés basés sur une méthode de développement formelle. La formalisation des diagrammes de relais à l'aide de bases mathématiques a pour objectif de permettre la modélisation de systèmes provenant de différents contextes, sur la base de l'adaptation du modèle mathématique, et la spécification formelle de ces systèmes dans différents langages de spécification formelle qui contiennent les mêmes bases mathématiques.

La formalisation de ces systèmes est divisée en modèles structurels et comportementaux. Le modèle structurel est une transcription de la structure du système modélisé en diagrammes de relais en un modèle mathématique basé sur la théorie des graphes. En ce qui concerne le modèle comportemental, il utilise la structure modélisée afin de déterminer la relation entre les états des composants en fonction de leurs types. Ces modèles peuvent être utilisés pour l'analyse du système, néanmoins, leur objectif principal est de soutenir la spécification formelle du système en fournissant une description formelle de la structure et du comportement du système.

Ensuite, sur la base de ces modèles, nous avons proposé des directives de transformation afin de spécifier formellement les systèmes modélisés. Ces directives s'attachent à présenter la manière dont chaque clause de la machine B peut être spécifiée sur la base des informations présentées dans le modèle structurel et comportemental. De plus, les diagrammes de relais sont utilisés pour compléter les informations, car les modèles formalisés manquent d'informations sur les transitions d'état. Dans ce travail, nous avons présenté comment l'étude de cas ITCS peut être modélisée et spécifiée, puis nous avons utilisé le vérificateur de modèle ProB afin de le vérifier en fonction d'une condition de sécurité.

Perspectives

Comme ce travail fournit une base formalisée pour la structure et le comportement des SEF, ses contributions peuvent être utiles pour la modélisation et la spécification formelle de systèmes provenant de différents contextes dans différents langages de spécification formelle. En outre, la mise en œuvre de ces systèmes et l'automatisation de la méthodologie complète sont également des possibilités intéressantes.

Dans ce contexte, le raffinement et la mise en œuvre des SEF basés sur des relais en tant que systèmes informatiques est une première perspective. Bien qu'il existe déjà diverses méthodologies dans la littérature pour le raffinement des systèmes spécifiés en B, nous visons quand même la définition d'une méthodologie spécifique de raffinement complète pour nos systèmes à base de relais. Cette méthodologie doit se concentrer sur la séparation des entrées et des sorties du système, qui seront les connexions, ou l'interface, entre l'ordinateur et les composants

externes. De plus, nous envisageons la création d'un outil pour automatiser la transformation des diagrammes de relais vers la spécification formelle et l'implémentation du système. Cet outil doit pouvoir utiliser les modèles présentés dans cette thèse comme base de cette transformation.

Une autre perspective est l'utilisation de la modélisation conceptuelle afin d'améliorer les modèles mathématiques et la spécification formelle en considérant les informations sur l'environnement du système. L'exécution des SEF dépend des relations implicites entre certains composants ainsi que de certains aspects environnementaux, comme la position des trains et l'extension des voies, qui sont des informations conceptuelles visualisées par le conducteur du train. Tous ces détails peuvent avoir un impact sur l'exécution du système, c'est pourquoi la vérification de la sécurité du système doit les prendre en considération.

Une autre perspective intéressante est la spécification formelle des systèmes à base de relais dans différents langages de spécification formelle avec des objectifs différents. Dans un futur proche, nous proposons l'utilisation des CSP (Communicating Sequential Processes) pour la spécification des SEF en tant que systèmes concurrents en temps réel. L'utilisation de ce langage pourrait nous permettre d'analyser les aspects de concurrence de ces systèmes, comme l'existence de problèmes de blocages, de Livelock ou de vivacité. Une comparaison entre B et CSP peut être effectuée, en présentant les forces et les faiblesses de chaque langage.

Une dernière perspective est l'utilisation de l'obfuscation des logiciels afin de compenser les éventuels problèmes de sécurité. Comme le travail actuel se concentre sur le maintien de la sûreté de fonctionnement du système, il peut également être intéressant de vérifier d'autres aspects de ces systèmes, comme la sécurité-confidentialité, par exemple. Les systèmes à base de relais sont naturellement sûrs du point de vue sécurité-confidentialité, tandis que le comportement des systèmes informatiques peut être plus facilement altéré par des attaquants mal intentionnés. Ainsi, une solution possible à ce problème est l'utilisation de l'obfuscation des logiciels, qui rend le système mis en œuvre plus difficile à lire et à comprendre par les humains, ce qui augmente la résistance aux attaques. Enfin, nous avons pour objectif de faire une analyse de l'insertion des techniques d'obfuscation logicielle dans la méthode de raffinement de la méthode B, fournissant une meilleure méthode formelle de développement de logiciels axée sur la sûreté et la sécurité.

Bibliography

- [J.-R. Abrial 2005] Abrial, J.-R. (2005). *The B-book: assigning programs to meanings*. Cambridge University Press (cit. on pp. 41, 153).
- [J.-R. Abrial and Hallerstede 2007] Abrial, J.-R. and S. Hallerstede (2007). “Refinement, decomposition, and instantiation of discrete models: Application to Event-B”. In: *Fundamenta Informaticae* 77.1-2, pp. 1–28 (cit. on p. 44).
- [J.-R. Abrial, Lee, et al. 1991] Abrial, J.-R., M. Lee, D. Neilson, P. Scharbach, and I. Sørensen (1991). “The B-method”. In: *International symposium of VDM Europe*. Springer, pp. 398–405 (cit. on pp. 2, 31, 41).
- [Akita et al. 1985] Akita, K., T. Watanabe, H. Nakamura, and I. Okumura (1985). “Computerized interlocking system for railway signaling control: SMILE”. In: *IEEE Transactions on Industry applications* 3, pp. 826–834 (cit. on pp. 1, 17).
- [Almeida Pereira, Debbech, et al. 2019] Almeida Pereira, D. I. de, S. Debbech, M. Perin, P. Bon, and S. Collart-Dutilleul (2019). “Formal Specification of Environmental Aspects of a Railway Interlocking System Based on a Conceptual Model”. In: *International Conference on Conceptual Modeling*. Springer, pp. 338–351 (cit. on p. 146).
- [Almeida Pereira, Malki, et al. 2018] Almeida Pereira, D. I. de, O. Malki, P. Bon, M. Perin, and S. Collart-Dutilleul (2018). “An MDA approach for the specification of relay-based diagrams”. In: *International Conference on Model and Data Engineering*. Springer, pp. 17–29 (cit. on p. 145).
- [Amendola et al. 1996] Amendola, A., L. Impagliazzo, P. Marmo, G. Mongardi, G. Sartore, and A. Trasporti (1996). “Architecture and safety requirements of the ACC railway interlocking system”. In: *Proceedings of IEEE International Computer Performance and Dependability Symposium*. IEEE Comput. Soc. Press. DOI: 10.1109/ipds.1996.540195¹ (cit. on p. 58).
- [Ammann 2016] Ammann, P. (Dec. 2016). *Introduction to Software Testing*. Cambridge University Press. ISBN: 9781107172012. URL: <https://www.xarg.org/ref/a/1107172012/> (cit. on p. 14).
- [Antoni 2009] Antoni, M. (July 2009). “Formal validation method for computerized railway interlocking systems”. In: *2009 International Conference on Computers & Industrial Engineering*. IEEE. DOI: 10.1109/iccie.2009.5223968² (cit. on p. 57).
- [Banci, Fantechi, and Gnesi 2004] Banci, M., A. Fantechi, and S. Gnesi (2004). “The role of formal methods in developing a distributed railway interlocking system”. In: *Proc. of the 5th Symposium on Formal Methods for Automation and Safety in Railway and Automotive Systems (FORMS/FORMAT 2004)*. Technical University of Braunschweig, pp. 220–230 (cit. on pp. 48, 49, 57, 58).
- [Behm et al. 1999] Behm, P., P. Benoit, A. Faivre, and J.-M. Meynadier (1999). “METEOR: A successful application of B in a large project”. In: *International Symposium on Formal Methods*. Springer, pp. 369–387 (cit. on pp. 2, 25, 31, 44, 45).
- [Bellon 2014] Bellon, H. (2014). *Data Preparation Guide for Interlocking used in the Belgian Railways*. Tech. rep. (cit. on p. 60).
- [Ben Ayed, Collart-Dutilleul, and Prun 2016] Ben Ayed, R., S. Collart-Dutilleul, and E. Prun (2016). “Formal Methods To Tailored Solution For Single Track Low Traffic French Lines”. In: *International Railway Safety Council (IRSC)* (Paris, France) (cit. on p. 45).

¹<https://doi.org/10.1109/ipds.1996.540195>

²<https://doi.org/10.1109/iccie.2009.5223968>

- [Bienmüller, Damm, and Wittke 2000] Bienmüller, T., W. Damm, and H. Wittke (2000). “The StateMate verification environment”. In: *International Conference on Computer Aided Verification*. Springer, pp. 561–567 (cit. on p. 59).
- [Biere et al. 1999] Biere, A., A. Cimatti, E. Clarke, and Y. Zhu (1999). “Symbolic Model Checking without BDDs”. In: *Tools and Algorithms for the Construction and Analysis of Systems*. Springer Berlin Heidelberg, pp. 193–207. doi: 10.1007/3-540-49059-0_14³ (cit. on p. 58).
- [Bjørner 1979a] Bjørner, D. (1979a). “The vienna development method (VDM)”. In: *Mathematical Studies of Information Processing*. Springer, pp. 326–359 (cit. on p. 41).
- [Bjørner 1979b] — (1979b). “The vienna development method (VDM)”. In: *Mathematical Studies of Information Processing*. Springer Berlin Heidelberg, pp. 326–359. doi: 10.1007/3-540-09541-1_33⁴ (cit. on p. 57).
- [Bjørner 1998] — (1998). “The FME rail annotated rail bibliography”. In: (cit. on p. 48).
- [Black et al. 1996] Black, P. E., K. M. Hall, M. D. Jones, T. N. Larson, and P. J. Windley (1996). “A brief introduction to formal methods [hardware design]”. In: *Proceedings of Custom Integrated Circuits Conference*. IEEE, pp. 377–380 (cit. on p. 31).
- [Bondy and Murty 1976] Bondy, J. A. and U. S. R. Murty (June 1976). *Graph Theory With Applications*. Elsevier Science Ltd/North-Holland. ISBN: 0444194517. URL: <https://www.xarg.org/ref/a/0444194517/> (cit. on pp. 38, 39).
- [Booch, Bryan, and Petersen 1994] Booch, G., D. L. Bryan, and C. G. Petersen (1994). *Software engineering with Ada*. Vol. 30608. Addison-Wesley Professional (cit. on p. 45).
- [Borälv 1998] Borälv, A. (Apr. 1998). “Case Study: Formal Verification of a Computerized Railway Interlocking”. In: *Formal Aspects of Computing* 10.4, pp. 338–360. doi: 10.1007/s001650050021⁵ (cit. on p. 60).
- [Bowen and Stavridou 1993] Bowen, J. and V. Stavridou (1993). “Safety-critical systems, formal methods and standards”. In: *Software Engineering Journal* 8.4, pp. 189–209 (cit. on p. 44).
- [Buchheit et al. 2011] Buchheit, G., O. Malassé, N. Brinzei, J. Lalouette, and M. Walter (2011). “Evaluation des performances d’un axe ferroviaire en fonction des caractéristiques fiabilistes de ses systèmes de signalisations”. In: (cit. on p. 54).
- [Busard et al. 2015] Busard, S., Q. Cappart, C. Limbrée, C. Pecheur, and P. Schaus (June 2015). “Verification of railway interlocking systems”. In: *Electronic Proceedings in Theoretical Computer Science* 184, pp. 19–31. doi: 10.4204/eptcs.184.2⁶ (cit. on p. 60).
- [Butler and Hallerstedde 2007] Butler, M. and S. Hallerstedde (2007). “The Rodin formal modelling tool”. In: *FACS 2007 Christmas Workshop: Formal Methods in Industry*, pp. 1–5 (cit. on p. 44).
- [Cavada et al. 2018] Cavada, R., A. Cimatti, S. Mover, M. Sessa, G. Cadavero, and G. Scaglione (2018). “Analysis of Relay Interlocking Systems via SMT-based Model Checking of Switched Multi-Domain Kirchhoff Networks”. In: *2018 Formal Methods in Computer Aided Design (FMCAD)*. IEEE, pp. 1–9 (cit. on pp. 52, 53, 55).
- [Char et al. 1983] Char, B. W., K. O. Geddes, W. M. Gentleman, and G. H. Gonnet (1983). “The design of Maple: A compact, portable, and powerful computer algebra system”. In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 101–115. doi: 10.1007/3-540-12868-9_95⁷ (cit. on p. 59).
- [Cimatti et al. 1998a] Cimatti, A., F. Giunchiglia, G. Mongardi, D. Romano, F. Torielli, and P. Traverso (Apr. 1998a). “Formal Verification of a Railway Interlocking System using Model Checking”. In: *Formal Aspects of Computing* 10.4, pp. 361–380. doi: 10.1007/s001650050022⁸ (cit. on p. 60).

³https://doi.org/10.1007/3-540-49059-0_14

⁴https://doi.org/10.1007/3-540-09541-1_33

⁵<https://doi.org/10.1007/s001650050021>

⁶<https://doi.org/10.4204/eptcs.184.2>

⁷https://doi.org/10.1007/3-540-12868-9_95

⁸<https://doi.org/10.1007/s001650050022>

- [Cimatti et al. 1998b] — (1998b). “Model Checking Safety Critical Software with SPIN: an Application to a Railway Interlocking System”. In: *Computer Safety, Reliability and Security*. Springer Berlin Heidelberg, pp. 284–293. doi: 10.1007/3-540-49646-7_22⁹ (cit. on p. 60).
- [Clearsy 2011] Clearsy (2011). *Atelier B User Manual*. English. Version Version 4.0. ClearSy System Engineering. 110 pp. November 16, 2011 (cit. on p. 44).
- [Damm and Klose 2001] Damm, W. and J. Klose (2001). “Verification of a radio-based signaling system using the STATEMATE verification environment”. In: *Formal Methods in System Design* 19.2, pp. 121–141 (cit. on p. 59).
- [De Moura, Owre, and Shankar 2003] De Moura, L., S. Owre, and N. Shankar (2003). “The SAL language manual”. In: *Computer Science Laboratory, SRI International, Menlo Park, Tech. Rep. CSL-01-01* (cit. on p. 54).
- [Dipoppa et al. 2001] Dipoppa, G., G. D’Alessandro, R. Semprini, and E. Tronci (2001). “Integrating automatic verification of safety requirements in railway interlocking system design”. In: *Proceedings Sixth IEEE International Symposium on High Assurance Systems Engineering. Special Topic: Impact of Networking*. IEEE Comput. Soc. doi: 10.1109/hase.2001.966821¹⁰ (cit. on p. 58).
- [Elmstrøm, Larsen, and Lassen 1994] Elmstrøm, R., P. G. Larsen, and P. B. Lassen (1994). “The IFAD VDM-SL toolbox: a practical approach to formal specifications”. In: *ACM Sigplan Notices* 29.9, pp. 77–80 (cit. on pp. 58, 60).
- [Emerson 1990] Emerson, E. A. (1990). “Temporal and modal logic”. In: *Formal Models and Semantics*. Elsevier, pp. 995–1072 (cit. on p. 50).
- [CENELEC 2017a] CENELEC (Oct. 2017a). *Railway Applications - The Specification and demonstration of reliability, availability, maintainability and safety (RAMS) - Part 1: generic RAMS process*. Std. European Committee for Electrotechnical Standardisation (CENELEC) (cit. on pp. 16, 147, 148).
- [CENELEC 2017b] — (Oct. 2017b). *Railway Applications - The specification and demonstration of reliability, availability, maintainability and safety (RAMS) - Part 2: systems approach to safety*. Std. European Committee for Electrotechnical Standardisation (CENELEC) (cit. on p. 16).
- [CENELEC 2011] — (Oct. 2011). *Railway Applications - Communication, signalling and processing systems - Software for railway control and protection systems*. Std. European Committee for Electrotechnical Standardisation (CENELEC) (cit. on pp. 3, 4, 16, 25, 31, 149, 152).
- [CENELEC 2018] — (Nov. 2018). *Railway applications - Communication, signalling and processing systems - Safety related electronic systems for signalling*. Std. European Committee for Electrotechnical Standardisation (CENELEC) (cit. on pp. 16, 148).
- [Eris and Mutlu 2010] Eris, O. and I. Mutlu (Dec. 2010). “Design of signal control structures using formal methods for railway interlocking systems”. In: *2010 11th International Conference on Control Automation Robotics & Vision*. IEEE. doi: 10.1109/icarcv.2010.5707824¹¹ (cit. on p. 58).
- [Fantechi, Fokkink, and Morzenti 2013] Fantechi, A., W. Fokkink, and A. Morzenti (2013). “Some trends in formal methods applications to railway signaling”. In: *Formal Methods for Industrial Critical Systems*, pp. 61–84 (cit. on pp. 25, 31).
- [Ferlin et al. 2016] Ferlin, A., R. Ben-Ayed, P. Sun, S. Collart-Dutilleul, and P. Bon (2016). “Implementation of ERTMS: a methodology based on formal methods and simulation with respect to French national rules”. In: *Transportation Research Procedia* 14, pp. 1957–1966 (cit. on p. 45).
- [Ferrari et al. 2011] Ferrari, A., G. Magnani, D. Grasso, and A. Fantechi (2011). “Model Checking Interlocking Control Tables”. In: *FORMS/FORMAT 2010*. Springer Berlin Heidelberg, pp. 107–115. doi: 10.1007/978-3-642-14261-1_11¹² (cit. on p. 51).
- [Figueiredo and Perkusich n.d.] Figueiredo, J. de and A. Perkusich (n.d.). “Distributed control of track-vehicle system with fault-tolerant characteristics: a Petri net based approach”. In:

⁹https://doi.org/10.1007/3-540-49646-7_22

¹⁰<https://doi.org/10.1109/hase.2001.966821>

¹¹<https://doi.org/10.1109/icarcv.2010.5707824>

¹²https://doi.org/10.1007/978-3-642-14261-1_11

- 1995 *IEEE International Conference on Systems, Man and Cybernetics. Intelligent Systems for the 21st Century*. IEEE. doi: 10.1109/icsmc.1995.537788¹³ (cit. on p. 58).
- [Ghosh et al. 2016] Ghosh, S., A. Das, N. Basak, P. Dasgupta, and A. Katiyar (2016). “Formal methods for validation and test point prioritization in railway signaling logic”. In: *IEEE Transactions on Intelligent Transportation Systems* 18.3, pp. 678–689 (cit. on pp. 51, 56).
- [Gibson-Robinson et al. 2014] Gibson-Robinson, T., P. Armstrong, A. Boulgakov, and A. W. Roscoe (2014). “FDR3—a modern refinement checker for CSP”. In: *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, pp. 187–201 (cit. on pp. 51, 146).
- [Guiho and Hennebert 1990] Guiho, G. and C. Hennebert (1990). “SACEM software validation”. In: *Software Engineering, 1990. Proceedings., 12th International Conference on*. IEEE, pp. 186–191 (cit. on pp. 25, 31, 44).
- [Hall 1990] Hall, A. (Sept. 1990). “Seven myths of formal methods”. In: *IEEE Software* 7.5, pp. 11–19. doi: 10.1109/52.57887¹⁴ (cit. on p. 31).
- [Hansen 1994] Hansen, K. M. (1994). “Validation of a railway interlocking model”. In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 582–601. doi: 10.1007/3-540-58555-9_117¹⁵ (cit. on pp. 56, 58).
- [Hansen 1998a] — (1998a). “Formalising railway interlocking systems”. In: *Nordic Seminar on Dependable Computing Systems*. Citeseer, pp. 83–94 (cit. on pp. 1, 2, 17, 56, 58).
- [Hansen 1998b] — (1998b). “Modelling railway interlocking systems”. In: *Department of Computer Science, Technical University of Denmark*. Citeseer (cit. on pp. 57, 58).
- [Harel 1987] Harel, D. (1987). “Statecharts: A visual formalism for complex systems”. In: *Science of computer programming* 8.3, pp. 231–274 (cit. on p. 57).
- [Haxthausen 2014] Haxthausen, A. E. (2014). “Automated generation of formal safety conditions from railway interlocking tables”. In: *International journal on software tools for technology transfer* 16.6, pp. 713–726 (cit. on p. 54).
- [Haxthausen 2013] — (Dec. 2013). “Automated generation of formal safety conditions from railway interlocking tables”. In: *International Journal on Software Tools for Technology Transfer* 16.6, pp. 713–726. doi: 10.1007/s10009-013-0295-9¹⁶ (cit. on pp. 52–55).
- [Haxthausen, Kjær, and Le Bliguet 2011] Haxthausen, A. E., A. A. Kjær, and M. Le Bliguet (2011). “Formal development of a tool for automated modelling and verification of relay interlocking systems”. In: *International Symposium on Formal Methods*. Springer, pp. 118–132 (cit. on pp. 52–54).
- [Haxthausen, Le Bliguet, and Kjær 2008] Haxthausen, A. E., M. Le Bliguet, and A. A. Kjær (2008). “Modelling and verification of relay interlocking systems”. In: *Monterey Workshop*. Springer, pp. 141–153 (cit. on pp. 3, 24).
- [Hayes and Jones 1989] Hayes, I. J. and C. B. Jones (1989). “Specifications are not (necessarily) executable”. In: *Software Engineering Journal* 4.6, pp. 330–339 (cit. on p. 58).
- [Hei, Takahashi, and Hideo 2008] Hei, X., S. Takahashi, and N. Hideo (Sept. 2008). “Toward developing a Decentralized Railway Signalling System Using Petri Nets”. In: *2008 IEEE Conference on Robotics, Automation and Mechatronics*. IEEE. doi: 10.1109/ramech.2008.4681511¹⁷ (cit. on p. 57).
- [Hei, Takahashi, and Nakamura 2009] Hei, X., S. Takahashi, and H. Nakamura (2009). “Modeling and analyzing component-based distributed railway interlocking system with petri nets”. In: *IEEE Transactions on Industry Applications* 129.5, pp. 455–461 (cit. on p. 57).
- [Hei, Takahashi, Nakamura, et al. 2006] Hei, X., S. Takahashi, H. Nakamura, M. Fukuda, K. Iwata, and K. Sato (2006). “Improving Reliability of Railway Interlocking System with Component-based Technology (< Special Survey > Reliability and Maintainability of Informa-

¹³<https://doi.org/10.1109/icsmc.1995.537788>

¹⁴<https://doi.org/10.1109/52.57887>

¹⁵https://doi.org/10.1007/3-540-58555-9_117

¹⁶<https://doi.org/10.1007/s10009-013-0295-9>

¹⁷<https://doi.org/10.1109/ramech.2008.4681511>

- tion Systems)". In: *The Journal of Reliability Engineering Association of Japan* 28.8, pp. 557–568 (cit. on p. 57).
- [Hernando et al. 2012] Hernando, A., E. Roanes-Lozano, R. Maestre-Martinez, and J. Tejedor (Aug. 2012). "A logic-algebraic approach to decision taking in a railway interlocking system". In: *Annals of Mathematics and Artificial Intelligence* 65.4, pp. 317–328. DOI: 10.1007/s10472-012-9321-y¹⁸ (cit. on p. 60).
- [Hoare 1978] Hoare, C. A. R. (1978). "Communicating sequential processes". In: *Communications of the ACM* 21.8, pp. 666–677 (cit. on p. 50).
- [G. Holzmann 1997] Holzmann, G. (May 1997). "The model checker SPIN". In: *IEEE Transactions on Software Engineering* 23.5, pp. 279–295. DOI: 10.1109/32.588521¹⁹. URL: <https://doi.org/10.1109/32.588521> (cit. on p. 51).
- [G. J. Holzmann and Lieberman 1991] Holzmann, G. J. and W. S. Lieberman (1991). *Design and validation of computer protocols*. Vol. 512. Prentice hall Englewood Cliffs (cit. on p. 55).
- [ISO 2015] ISO (Sept. 2015). *Quality management systems — Requirements*. Std. International Organisation for Standardisation (ISO) (cit. on p. 16).
- [James, Lawrence, et al. 2013] James, P., A. Lawrence, F. Moller, M. Roggenbach, M. Seisenberger, A. Setzer, K. Kanso, and S. Chadwick (2013). "Verification of solid state interlocking programs". In: *International Conference on Software Engineering and Formal Methods*. Springer, pp. 253–268 (cit. on pp. 52, 54, 55).
- [James, Moller, et al. 2014] James, P., F. Moller, H. N. Nguyen, M. Roggenbach, S. Schneider, and H. Treharne (2014). "Techniques for modelling and verifying railway interlockings". In: *International Journal on Software Tools for Technology Transfer* 16.6, pp. 685–711 (cit. on p. 59).
- [Janota 2000] Janota, A. (2000). "Using Z specification for railway interlocking safety". In: *Periodica Polytechnica Transportation Engineering* 28.1-2, pp. 39–53 (cit. on p. 60).
- [Janschek 2011] Janschek, K. (2011). *Mechatronic systems design: methods, models, concepts*. Springer Science & Business Media (cit. on p. 55).
- [Jensen 1987] Jensen, K. (1987). "Coloured Petri Nets". In: *Petri Nets: Central Models and Their Properties*. Springer Berlin Heidelberg, pp. 248–299. DOI: 10.1007/978-3-540-47919-2_10²⁰. URL: https://doi.org/10.1007/978-3-540-47919-2_10 (cit. on pp. 46, 50).
- [Khan and N. A. Zafar 2009] Khan, S. A. and N. A. Zafar (Feb. 2009). "Towards the formalization of railway interlocking system using Z-notations". In: *2009 2nd International Conference on Computer, Control and Communication*. IEEE. DOI: 10.1109/ic4.2009.4909202²¹ (cit. on p. 60).
- [Khan, N. A. Zafar, et al. 2014] Khan, S. A., N. A. Zafar, F. Ahmad, and S. Islam (Jan. 2014). "Extending Petri net to reduce control strategies of railway interlocking system". In: *Applied Mathematical Modelling* 38.2, pp. 413–424. DOI: 10.1016/j.apm.2013.06.002²² (cit. on pp. 50, 57).
- [Knight 2002] Knight, J. C. (2002). "Safety critical systems". In: *Proceedings of the 24th international conference on Software engineering - ICSE '02*. ACM Press, pp. 547–550. DOI: 10.1145/581339.581406²³ (cit. on pp. 1, 14).
- [Korečko and Sobota 2014] Korečko, Š. and B. Sobota (2014). "Petri nets to B-language transformation in software development". In: *Acta polytechnica hungarica* 11.6, pp. 187–206 (cit. on p. 124).
- [Lalouette et al. 2010] Lalouette, J., R. Caron, F. Scherb, N. Brinzei, J.-F. Aubry, and O. Malassé (2010). "Evaluation des performances du système de signalisation ferroviaire européen superposé au système français, en présence de défaillances Performance assessment of European railway signalling system superposed of the French system in the presence of failures". In: (cit. on p. 54).

¹⁸<https://doi.org/10.1007/s10472-012-9321-y>

¹⁹<https://doi.org/10.1109/32.588521>

²⁰https://doi.org/10.1007/978-3-540-47919-2_10

²¹<https://doi.org/10.1109/ic4.2009.4909202>

²²<https://doi.org/10.1016/j.apm.2013.06.002>

²³<https://doi.org/10.1145/581339.581406>

- [Lecomte, Servat, Pouzancre, et al. 2007] Lecomte, T., T. Servat, G. Pouzancre, et al. (2007). “Formal methods in safety-critical railway systems”. In: *10th Brazilian symposium on formal methods*, pp. 29–31 (cit. on pp. 2, 25, 31, 45).
- [Leuschel and Butler 2008] Leuschel, M. and M. Butler (2008). “ProB: an automated analysis toolset for the B method”. In: *International Journal on Software Tools for Technology Transfer* 10.2, pp. 185–203 (cit. on p. 44).
- [Liu, Saat, and Barkan 2012] Liu, X., M. R. Saat, and C. P. L. Barkan (Jan. 2012). “Analysis of Causes of Major Train Derailment and Their Effect on Accident Rates”. In: *Transportation Research Record: Journal of the Transportation Research Board* 2289.1, pp. 154–163. doi: 10.3141/2289-20²⁴ (cit. on p. 15).
- [Marsan and Chiola 1987] Marsan, M. A. and G. Chiola (1987). “On Petri nets with deterministic and exponentially distributed firing times”. In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 132–145. doi: 10.1007/3-540-18086-9_23²⁵ (cit. on p. 58).
- [Martinez et al. in press] Martinez, S., D. I. de Almeida Pereira, P. Bon, S. Collart-Dutilleul, and M. Perin (in press). “Towards Safe and Secure Computer Based Railway Interlocking systems”. In: *International Journal of Transport Development and Integration* (cit. on p. 148).
- [McDonald and Anton 2001] McDonald, J. and J. Anton (2001). “SPECWARE-producing software correct by construction”. In: (cit. on p. 44).
- [Mirabadi and Yazdi 2009] Mirabadi, A. and M. Yazdi (2009). “Automatic generation and verification of railway interlocking control tables using FSM and NuSMV”. In: *Transport Problems* 4, pp. 103–110 (cit. on pp. 50, 51).
- [Mongardi 1993] Mongardi, G. (1993). “Dependable Computing for Railway Control Systems”. In: *Dependable Computing for Critical Applications 3*. Springer Vienna, pp. 255–277. doi: 10.1007/978-3-7091-4009-3_11²⁶ (cit. on p. 60).
- [Moura, Owre, and Shankar 2003] Moura, L. de, S. Owre, and N. Shankar (2003). *The SAL Language Manual*. Tech. rep. 333 Ravenswood Ave. Menlo Park, CA 94025 (cit. on p. 55).
- [Pasquale et al. 2003] Pasquale, T., E. Rosaria, M. Pietro, O. Antonio, and A. S. Ferroviario (2003). “Hazard analysis of complex distributed railway systems”. In: *22nd International Symposium on Reliable Distributed Systems, 2003. Proceedings*. IEEE, pp. 283–292 (cit. on pp. 2, 17).
- [Peterson 1977] Peterson, J. L. (1977). “Petri nets”. In: *ACM Computing Surveys (CSUR)* 9.3, pp. 223–252 (cit. on pp. 25, 45, 50).
- [Petri 1962] Petri, C. (1962). *Kommunikation mit Automaten*. Schriften des Rheinisch-Westfälischen Institutes für Instrumentelle Mathematik an der Universität Bonn. Technische Hochschule, Darmstadt. URL: <https://books.google.fr/books?id=NCZMvAEACAAJ> (cit. on p. 31).
- [Rétiveau 1987] Rétiveau, R. (1987). *La signalisation ferroviaire*. Presse de l’école nationale des Ponts et Chaussées (cit. on pp. 15, 19, 66).
- [Roanes-Lozano et al. 2011] Roanes-Lozano, E., A. Hernando, J. A. Alonso, and L. M. Laita (Sept. 2011). “A logic approach to decision taking in a railway interlocking system using Maple”. In: *Mathematics and Computers in Simulation* 82.1, pp. 15–28. doi: 10.1016/j.matcom.2010.05.024²⁷ (cit. on pp. 57, 59).
- [Rushby 1995] Rushby, J. (1995). “Formal specification and verification for critical systems: Tools, achievements, and prospects”. In: *Advances in Ultra-Dependable Distributed Systems*, pp. 282–296 (cit. on pp. 30, 31).
- [Schneider 2000] Schneider, S. (2000). *Concurrent and Real-time systems*. John Wiley and Sons (cit. on pp. 25, 31, 50, 146).
- [Schneider 2001] — (Oct. 2001). *The B-Method (Cornerstones of Computing)*. Red Globe Press. ISBN: 033379284X. URL: <https://www.xarg.org/ref/a/033379284X/> (cit. on pp. 32, 41, 43).

²⁴<https://doi.org/10.3141/2289-20>

²⁵https://doi.org/10.1007/3-540-18086-9_23

²⁶https://doi.org/10.1007/978-3-7091-4009-3_11

²⁷<https://doi.org/10.1016/j.matcom.2010.05.024>

- [Schneider and Treharne 2005] Schneider, S. and H. Treharne (2005). “CSP theorems for communicating B machines”. In: *Formal Aspects of Computing* 17.4, pp. 390–422 (cit. on p. 59).
- [Schön et al. 2013] Schön, W., G. Larraufie, G. Moëns, and J. Poré (Apr. 2013). *Railway Signalling and Automation*. Vol. 1. LA VIE DU RAIL. ISBN: 2918758485. URL: <https://www.xarg.org/ref/a/2918758485/> (cit. on pp. 14, 66).
- [Schön et al. 2014] — (Jan. 2014). *Railway Signalling and Automation*. Vol. 3. LA VIE DU RAIL. ISBN: 2918758647. URL: <https://www.xarg.org/ref/a/2918758647/> (cit. on pp. 18, 19, 66).
- [She et al. 2007] She, X., Y. Sha, Q. Chen, and J. Yang (June 2007). “The Application of Graphic Theory on Railway Yard Interlocking Control System”. In: *2007 IEEE Intelligent Vehicles Symposium*. IEEE. DOI: 10.1109/ivs.2007.4290228²⁸ (cit. on pp. 49, 50).
- [Spivey and J. Abrial 1992] Spivey, J. M. and J. Abrial (1992). *The Z notation*. Prentice Hall Hemel Hempstead (cit. on pp. 25, 41, 59).
- [Sun 2015] Sun, P. (2015). “Model based system engineering for safety of railway critical systems”. PhD thesis. Ecole Centrale de Lille (cit. on pp. 31, 52–58).
- [Sun, Collart-dutilleul, and Bon 2015] Sun, P., S. Collart-dutilleul, and P. Bon (June 2015). “A model pattern of railway interlocking system by Petri nets”. In: *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE. DOI: 10.1109/mtits.2015.7223292²⁹ (cit. on pp. 25, 46, 52, 54–58).
- [Suppes 1999] Suppes, P. (1999). *Introduction to logic*. Mineola, N.Y: Dover Publications. ISBN: 978-0-486-40687-9 (cit. on pp. 31, 33).
- [Theeg 2017] Theeg, G. (2017). *Railway signalling & interlocking international compendium*. BingenHamburg: PMC Media House GmbH. ISBN: 3962451560 (cit. on pp. 12, 14, 17, 25).
- [Trudeau 1994] Trudeau, R. J. (Feb. 1994). *Introduction to Graph Theory (Dover Books on Mathematics)*. Dover Publications. ISBN: 0486678709. URL: <https://www.xarg.org/ref/a/0486678709/> (cit. on p. 38).
- [Van Eijk 1997] Van Eijk, P. (1997). “Verifying relay circuits using state machines”. In: *Logic Group Preprint Series* 173 (cit. on pp. 52–55).
- [Vanit-Anunchai 2010] Vanit-Anunchai, S. (2010). “Modelling Railway Interlocking Tables Using Coloured Petri Nets”. In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 137–151. DOI: 10.1007/978-3-642-13414-2_10³⁰ (cit. on p. 50).
- [Winter 2002] Winter, K. (2002). “Model checking railway interlocking systems”. In: *Australian Computer Science Communications*. Vol. 24. Australian Computer Society, Inc., pp. 303–310 (cit. on pp. 25, 50, 51).
- [Winter et al. 2006] Winter, K., W. Johnston, P. Robinson, P. Strooper, and L. Van Den Berg (2006). “Tool support for checking railway interlocking designs”. In: *Proceedings of the 10th Australian workshop on Safety critical systems and software-Volume 55*. Australian Computer Society, Inc., pp. 101–107 (cit. on pp. 50, 51).
- [Woodcock and Davies 1996] Woodcock, J. and J. Davies (1996). “Using Z: Specification flRefinement fland Proof”. In: (cit. on pp. 30, 31).
- [Woodcock, Larsen, et al. 2009] Woodcock, J., P. G. Larsen, J. Bicarregui, and J. Fitzgerald (2009). “Formal methods: Practice and experience”. In: *ACM computing surveys (CSUR)* 41.4, pp. 1–36 (cit. on pp. 31, 44).
- [Xiangxian, Yulin, and hai 2011] Xiangxian, C., H. Yulin, and H. hai (May 2011). “A component-based topology model for railway interlocking systems”. In: *Mathematics and Computers in Simulation* 81.9, pp. 1892–1900. DOI: 10.1016/j.matcom.2011.02.007³¹ (cit. on pp. 49, 56).
- [N. Zafar 2006] Zafar, N. (2006). “Formal model for moving block railway interlocking system based on un-directed topology”. In: *2006 International Conference on Emerging Technologies*. IEEE. DOI: 10.1109/icet.2006.335983³² (cit. on p. 60).

²⁸<https://doi.org/10.1109/ivs.2007.4290228>

²⁹<https://doi.org/10.1109/mtits.2015.7223292>

³⁰https://doi.org/10.1007/978-3-642-13414-2_10

³¹<https://doi.org/10.1016/j.matcom.2011.02.007>

³²<https://doi.org/10.1109/icet.2006.335983>

- [N. A. Zafar 2006] Zafar, N. A. (2006). "Modeling and formal specification of automated train control system using Z notation". In: *2006 IEEE International Multitopic Conference*. IEEE, pp. 438–443 (cit. on pp. 25, 59).
- [N. A. Zafar 2009] — (2009). "Formal specification and validation of railway network components using Z notation". In: *IET software* 3.4, pp. 312–320 (cit. on p. 59).
- [N. A. Zafar, Khan, and Araki 2012] Zafar, N. A., S. A. Khan, and K. Araki (2012). "Towards the safety properties of moving block railway interlocking system". In: *Int. J. Innovative Comput., Info & Control* 8.7, pp. 5677–5690 (cit. on p. 60).
- [Zhang 1997] Zhang, H. (1997). "SATO: An efficient propositional prover". In: *International Conference on Automated Deduction*. Springer, pp. 272–275 (cit. on p. 58).
- [Zimmermann and Hommel n.d.] Zimmermann, A. and G. Hommel (n.d.). "A train control system case study in model-based real time system design". In: *Proceedings International Parallel and Distributed Processing Symposium*. IEEE Comput. Soc. doi: 10.1109/ipdps.2003.1213234³³ (cit. on p. 58).

³³<https://doi.org/10.1109/ipdps.2003.1213234>

Contents

General Introduction	1
I Preliminary	9
1 Background	11
1.1 Introduction	12
1.2 Railway Systems and Signalling	12
1.2.1 Safety-critical Aspects in the Railway Field	14
1.2.2 Relay-based and Computer-controlled RIS	17
1.2.3 Case Study	26
1.3 Formal Methods and Mathematical Foundations	30
1.3.1 Formal Specification Methodologies	30
1.3.2 Propositional and First Order Logic	31
1.3.3 Basics of Set Theory and Relations	34
1.3.4 Graph Description Based on Set Theory	38
1.3.5 The B-method	41
2 Formal Specification of Railway Interlocking Systems	47
2.1 Introduction	48
2.2 Timetables-based Approaches	49
2.3 Relay and Computer-based RIS Formal Specification	51
2.3.1 Formalising Relay-based RIS Logic	52
2.3.2 Formal Specification and Implementation of Computer-based RIS	56
2.3.3 Other approaches	59
2.4 Conclusions	61
II Methodology	63
3 Formalisation of Relay-based RIS: A Graph Approach	65
3.1 Introduction	66
3.2 RIS Basic Logical Description	67
3.2.1 Relay Diagrams Basic Structure	67
3.2.2 Electrical Components Structural and Behavioural Formalisation	70
3.2.3 Timed Blocks Impact on the System State Definition	81
3.2.4 Capacitors and their Impact on the System State Definition	86
3.3 Flashing Lights: An Energy Source Variation	94
3.4 Formalisation Support for the System Verification	97
3.4.1 Structural Well-definedness Verification	97
3.4.2 Behavioural Safety Conditions Definition	100
3.5 Case Study Specification and Analysis	102
3.5.1 Structural Formalisation	102

3.5.2 Behavioural Formalisation and Verification	109
3.6 Discussion	113
3.7 Formal Specification Based on the Formalisation	114
4 RIS B Formal Specification: A Diagram-specific Approach	117
4.1 Introduction	118
4.2 Behavioural Specification Based on the System State Space	119
4.2.1 System Variables and State-space Organisation	119
4.2.2 State Evolution Specification	124
4.3 Discussion	136
III Conclusions	141
Conclusions and Perspectives	143
Résumé Étendu en Français	149
Bibliography	159

Abstract

Relay-based Railway Interlocking Systems (RIS) are critical systems and must be specified and safety proved in order to guarantee the absence of hazards during their execution. However, this is a challenging task, since Relay-based RIS are generally only structurally modelled in a way that their behavioural analysis are made manually based on the experts knowledge about the system. Thus, the existence of a RIS behavioural formal description is imperative in order to be able to perform safety proofs. Furthermore, as Computer-based RIS tend to be less expensive, more maintainable and extendable, the industry has interest in the existence of a methodology for transforming the existing Relay-based RIS into Computer-based RIS.

Formal specification methodologies are grounded in strong mathematical foundations that allow the systems safety proof. Besides, many formal specification languages support not only the verification, but also the implementation of these systems through a formal development process. Thus, Formal Methods may be the key in order to prove the RIS safety and implement them with computer-based technologies.

This thesis addresses two main propositions. Firstly, it presents an analysis of the relay diagrams information and a formalisation of the Relay-based RIS structure and behaviour based on mathematical expressions as a way to create a certain level of formalisation of the systems. The resulting model can be extended and adapted in order to conform to different railway contexts and it can be used in order to support the specification of these systems in different formal specification languages. Then, this thesis presents how the RIS formal model can be adapted in order to formally specify these systems in the B-method, a formal specification language with a successful history in the railway field, which allows the system safety proof and implementation as computer-based systems.

As a result, this thesis presents a complete methodology for the specification and verification of Relay-based Railway Interlocking Systems, giving support for the systems safety proof in different contexts and for their specification and implementation in many different formal languages.

Keywords: Railway Interlocking Systems, Relay Diagrams, Formal Methods, B-method, Formal Specification, Logical Models

ANALYSE ET SPÉCIFICATION FORMELLE DES SYSTÈMES D'ENCLÈCHEMENT FERROVIAIRE BASÉS SUR LES RELAIS

Résumé

Les Systèmes d'Enclenchement Ferroviaire (SEF) basés sur des relais sont des systèmes critiques, ils doivent être spécifiés et leur sécurité doit être prouvée afin de garantir l'absence de dangers lors de leurs exécutions. Toutefois, il s'agit d'une tâche difficile, car les SEF à relais ne sont généralement modélisés que de manière structurelle, de sorte que leur analyse comportementale est effectuée manuellement sur la base des connaissances des experts sur le système. Cependant, l'existence d'une description formelle du comportement des SEF est impérative pour pouvoir effectuer des preuves de sécurité. En outre, comme les SEF informatisés ont tendance à être moins chers, plus faciles à entretenir et à faire évoluer, le secteur ferroviaire a intérêt à ce qu'il existe une méthodologie pour transformer des SEF à relais existants en SEF informatisés.

Les méthodologies formelles de spécification sont fondées sur des bases mathématiques solides qui permettent de prouver la sécurité des systèmes. En outre, de nombreux langages de spécification formelle prennent en charge non seulement la vérification, mais aussi la mise en œuvre de ces systèmes par un processus de développement formalisé. Ainsi, les méthodes formelles peuvent être la clé pour prouver la sécurité des SEF et les mettre en œuvre en utilisant des technologies informatiques.

Cette thèse aborde deux propositions principales. Premièrement, elle présente une analyse des informations des diagrammes à relais et de la formalisation de la structure et du comportement des SEF basés sur des expressions mathématiques afin de créer un certain niveau de formalisation des systèmes. Le modèle résultant peut être étendu et adapté afin de se conformer à différents contextes ferroviaires et il peut aussi être utilisé afin de soutenir la spécification de ces systèmes dans différents langages de spécification formels. Ensuite, cette thèse présente comment le modèle formel des SEF peut être adapté afin de spécifier formellement ces systèmes selon la méthode B, un langage de spécification formel qui a déjà été utilisé avec succès dans le domaine ferroviaire et qui permet de prouver la sécurité du système et de le mettre en œuvre en tant que système informatique.

En définitive, cette thèse présente une méthodologie complète pour la spécification et la vérification des Systèmes d'Enclenchement Ferroviaire basés sur des relais, en fournissant un support pour la preuve des systèmes dans différents contextes et pour leur spécification et leur mise en œuvre dans de nombreux langages formels différents.

Mots clés : Systèmes d'Enclenchement Ferroviaire, Diagrammes de Relais, Méthodes Formelles, Méthode B, Spécification Formelle, Modèles Logiques