Secure and Reliable Smart Cyber-Physical Systems

Samir Ouchani

HESAM UNIVERSITÉ

le cnam

Habilitation à Diriger des Recherches in Computer Science

Secure and Reliable Smart Cyber Physical Systems

Presented by: Samir OUCHANI
Enseignant Chercheur, CESI Engineering School - LINEACT CESI, Aix-en-Provence

Friday, November 18th, 2022
at CESI, Lyon Campus, Lyon

Jury

Mrs. Hanifa BOUCHENEB	Professor, GIGL, Polytechnique Montréal	Reporter
Mr. Mohamed MOSBAH	Professor, LABRI, University of Bordeaux	Reporter
Mr. David DELAHAYE	Professor, LIRMM, University of Montpellier	Reporter
Mrs. Parisa GHODOUS	Professor, LIRIS, University of Lyon 1	Examiner
Mr. Grabriele LENZINI	Professor, SnT, University of Luxembourg	Examiner
Mr. Kamel BARKAOUI	Professor, CEDRIC, CNAM Paris	Examiner
Mr. David BAUDRY	CESI Research Director, LINEACT CESI, Rouen	Examiner
Mr. Yohan DUPUIS	CESI Research Director, LINEACT CESI, Rouen	Guarantor

Acknowledgments

To you and only you.

Résumé

Les systèmes cyber-physiques intelligents (**SCPS**) sont des entités hétérogènes autonomes et interconnectées. Ils jouent un rôle central dans les infrastructures critiques. Ces systèmes sont devenus davantage dépendants des différentes technologies de communication embarquées. La pluralité de ces modules de communication augmente considérablement les possibilités d'attaques et rend les **SCPS** plus sensibles. En effet, ils peuvent engendrer de nouvelles catégories de vulnérabilités, ce qui pourrait entraîner des dommages importants. Mes travaux de recherche visent à développer des solutions garantissant les exigences fonctionnelles, la sécurité et la résilience des **SCPS**. Ma production scientifique s'appuie, entre autres, sur les méthodes formelles et les techniques de l'intelligence artificielle afin d'assurer la fiabilité la sécurité de ces systèmes.

Mots-clés : Sécurité, Fiabilité, Systèmes Cyber-Physiques, Protocoles légers, Blockchain, Méthodes Formelles, Intélligence Artificielle.

Abstract

Smart Cyber-Physical Systems (**SCPS**) are heterogeneous inter-operable autonomous entities that play a crucial role in critical infrastructures. And as a result of supporting novel communication and remote control features, they became more dependent on connectivity, which increased attack surfaces and introduced errors that elevated the likelihood of **SCPS** errors. Additionally, they may cause new types of errors, faults, and vulnerabilities, leading to significant economic damage. In my ongoing and future research, I aim to develop scalable solutions that satisfy the systems' requirements, ensure security, and support resilience and synergy between the components of **SCPS**. My research activities involve the application of software engineering methodologies and artificial intelligence techniques to solve critical societal and industrial problems related to **SCPS**.

Keywords: Security, Reliability, Cyber-Physical System, Light-weight Protocols, Blockchains, Formal Methods, Artificial Intelligence.

Contents

Chapter 1

Introduction

Content

Samir Ouchani holds a Ph.D. in Computer Science from Concordia University in Montreal, Canada, specializing in Computer Security, Software Engineering, and Formal Methods. My research activities primarily involve developing techniques and methodologies to strengthen cybersecurity and detect vulnerabilities through formal methods, cryptography, and data mining. Typically, these techniques are used for complex and large interdisciplinary systems, such as cyber-physical systems that contain interconnected objects. My academic and industrial experience have given me a strong aptitude for integrating into international institutions and high-level research teams such as Ericsson research (Montreal, Canada), Security and Trust (Luxembourg), and Verimag (Grenoble, France). I currently teach courses in security, formal methods, networking, software engineering, web development, and object-oriented coding to diverse audiences. At CESI, an engineering school in Aix-en-Provence, France, I am also responsible for cyber security and data science courses. My research work is valued at the level of the scientific community by numerous publications in internationally renowned journals and through my involvement in several international events. In this document, I describe most of my research activities, including the ongoing and near-future research projects.

1.1 Context

The complexity of modern systems, especially cyber-physical ones, is caused by the tight integration and interaction of various components, which may comprise a variety of features: computational, networking, social, and physical. Specifically, *Smart-Cyber-Physical-Systems* (**SCPS**) are complex

and heterogeneous systems that control large physical structures and entities by executing optimized strategies and making smart decisions. **SCPS** are the "controllers" of the interactions between communication networks and the vehicles in railway, automotive and avionic; they are the "brain" in smart buildings, and the "safety patrols" in civil physical infrastructures (e.g. power grids, water-waste systems) or digital infrastructures (integrated healthcare systems, communication grid). **SCPS** integrate physical elements (e.g. objects, sensors), informational elements (policies, norms, data), and computational elements (e.g. procedure, communication protocols). **SCPS** are critical cogs in critical infrastructures. In railway control, they are responsible, among others, of the safe functioning of railway switches, semaphores and levels crossing, including all the consoles and processes used for this purpose. In a smart building, they supervise the critical infrastructure's operations, e.g. they monitor the physical access to rooms, control the alarm system, and supervise other ordinary processes such as the blinding of windows and the heating of rooms.

It should be evident that **SCPS** must be highly dependable and trustworthy. Ensuring dependability is a challenging task in itself, and well studied elsewhere [1]; but there is an additional quality that, today, has become for **SCPS** as important as dependability: *security*. **SCPS** are in fact highly interconnected, networked, and open systems. They operate not in isolation from the physical, digital, and social environments where they are displaced; on the contrary, they receive stimuli continuously from those environments, and despite that they have to be robust and resilient to misuses, abuses, and intrusions. Often, **SCPS** incorporate elaborated mechanisms to ensure security. For instance, **SCPS** employed in railway control have consoles and terminals secured into buildings arrayed along the railways that prevent intrusions by employing defense artifacts:– picking-up the building door's lock, triggers an electric shock; failing repeatedly to log-in to the indoor terminal, causes a blackout in the building, the automatic locking of all exits, and the exhalation of soporific gas.

1.2 Challenges

SCPS should be secure against the largest set of possible attacks; but this quality is not easily assessable. The analysis of security is a task that, in other domains, has been pursued well with the use of *formal methods* [2]. Especially, they have worked successfully in the analysis of Internet protocols [3], and they promise to be as well helpful for the analysis of the security of **SCPS**. However, this is a just borne research topic, and no established solution is mature yet. Several challenges have to be addressed before adapting the formal method toolkit and using them in the **SCPS**'s security analysis.

- To include all the heterogeneous elements that compose a **SCPS**. This heterogeneity is a challenge for those who aim to model **SCPS**'s functioning and analyze their security and ensure their reliability. Different aspects need to be combined consistently into a single model e.g. the **SCPS**' physical features and controls, its digital data and processes, its analogical controls, its

constraint policies, and the interaction among all these elements as well as the interactions with the environments. Today, there are no solutions that offer a unified approach to model and analyze a **SCPS** security and reliability, and that consider a **SCPS**'s heterogeneous elements together.

- To consider the socio-technical dimension of **SCPS**, i.e. the **SCPS**'s interaction with people. As well as the physical element, people can ensure or compromise security, but no **SCPS**' models include socio-technical aspects so far.

- To include realistic threats models. In **SCPS**, an adversary that tries to corrupt the elements on which the security of **SCPS** depends upon can resort to strategies of social engineering and similar deceptive tricks. They are missing in the model commonly used in symbolic formal approach, the Dolev-Yao intruder [4]. This last, despite has proved itself to be the most powerful adversary model against which one can quest for breaching security protocols. It may not be the most model when the socio-technical security steps in [5]: from one side, it misses the ability to interfere with humans, such as the social-engineering skills; from another, it is too powerful since, for example, it can listen to any communications. This ability is unrealistic when the channel is visual since to eavesdrop on communication, the adversary must be, in this last example, in the visual proximity of one of the peers.

- To perform an analysis of security that is more informative than the worst-case analysis, the usual analysis in protocol security. The traditional approach often shows that a system is insecure since very rarely are complex systems invulnerable. But one would like to understand other parameters to qualify the discovered attacks better. For example, one would like to know the cost required to execute the attack. No formal approach offers today a different analysis than the worst-case one. Besides, the traditional verification provides a single counter-example, as this is sufficient to show that the system under verification can be violated. However, in a **SCPS**'s safety assessment, it is necessary to highlight all possible combinations of failures that lead to losing a function.

- It is known that **SCPS** generate and deal with a considerable amount of heterogeneous data, and many data techniques should be leveraged to **SCPS** applications. Merging formal methods and data mining techniques to deal with **SCPS** complexity in terms of modeling and analysis is an interesting topic to be investigated.

It has been shown in the literature that security-related safety assessment activities are different from the traditional verification ones where formal methods and data analysis techniques are used. Further, a lack of a unified modeling language able to model **SCPS** component and the way they interact. This research is about developing languages, techniques, and tools to support data and formal analysis of security and reliability for **SCPS**. It intends to study formal calculi to model the

variegate components of **SCPS** and to provide algorithms and tools for an automatic or computer-assisted analysis of the **SCPS** security, including its socio-technical and environmental dimensions.

1.3 Objectives

As will be presented in the next chapters, my research activities look to enhance security verification techniques by mixing formal methods and data mining approaches with a focus on their application on emergent systems, especially smart-cyber-physical systems since they are widely deployed in our daily-life needs. The goal is to develop practical and theoretical solutions that enable security assessment and resiliency reinforcement for **SCPS** by achieving the following objectives.

- To capture the underlying concepts of **SCPS** with an adequate semantics, the first objective of is to categorize and improve the existing modeling techniques and stereotype them for **SCPS** in order to produce a standardized **SCPS** modeling approach.
- Developing a sound and automatic framework, by relying on formal methods together with machine learning techniques, to ensure the correctness and gauge how well a **SCPS** is secure and resilient. In addition, it will result in improving the resilience of **SCPS** against dynamic environment conditions and cyber-attacks by developing a smart and dynamic recommendation system.
- Proposing optimal and scalable techniques for the previous objectives, and enabling a large library of security and resilience templates, including: security properties, policies, attacks, recovering plans and countermeasures.
- Implementing and releasing software supporting the developed methodology. Also, validating and testing the proposed concepts and tools on real-life industrial case studies.

1.4 Organization

Next chapters present the achieved results, and also our current research and projects in perspective. Chapter 2 shows our results about the developed semantics of **SCPS**, and Chapter 3 presents how security are reliability are specified and modeled in **SCPS**. Then, Chapter 4 gives different developed tools and approaches to enhance the analysis of **SCPS**. Finally, our research projections are presented in Chapter 5.

Chapter 2

SCPS Semantics

Content

In general, cyber-physical systems are difficult to model and this is due to many varieties and aspects of their components. To dead with the modeling complexity of CPS, we have developed a semantic for SysML (System Modeling Language) diagrams [6], since it is de facto modeling graphical language dedicated to industrial. Further, we have presented an algebraic specification to Cyber-Physical systems with a social dimension called Socio-Cyber-Physcial systems [7] and [8]. Furthermore, since priority is an important property to design and ensure the correct deployment of IoT (Internet o Things) based systems, we derived a formal semantics to Behavior Interaction Priority (BIP) analysis tool [9]. This chapter is based on the following contributions.

1. Abdelhakim Baouya, O. A. Mohamed, **Samir Ouchani**, and D. Bennouar, "Reliability-driven Automotive Software Deployment based on a Parametrizable Probabilistic Model Checking," *Expert Systems with Applications*, p. 114572, 2021.

2. **Ouchani, Samir**, "A security policy hardening framework for socio-cyber-physical systems," *Journal of Systems Architecture*, vol. 119, p. 102259, 2021.

3. Dahmane Walid, Miloud, **Ouchani, Samir**, and H. Bouarfa, "Towards a reliable smart city through formal verification and network analysis," *Computer Communications*, vol. 180, pp. 171–187, 2021.

4. Baouya, Abdelhakim, O. A. Mohamed, D. Bennouar, and **Ouchani, Samir**, "Safety analysis of train control system based on model-driven design methodology," *Computers in Industry*, vol. 105, pp. 1–16, 2019.

5. Baouya, Abdelhakim, S. Chehida, **Ouchani, Samir**, S. Bensalem, and M. Bozga, "Generation and verification of learned stochastic automata using k-nn and statistical model checking," *Applied Intelligence*, Nov 2021.

2.1 System Modeling Language

To model a system, we considered SysML activity diagrams to design its behaviour and parametric diagrams to specify its structure parameters and variables [11].

In our formalism [12], we denote by $\mathtt{a}$, $\mathtt{a}_i$, $\mathtt{a}_t$, and $\mathtt{a}_f$ opaque (atomic), initial, flow final, and activity final nodes, respectively; $\mathtt{a} \blacktriangleright v$ and $\mathtt{a} \blacktriangleleft v$ to send and receive a value v by the node $\mathtt{a}$, $\mathtt{a} \uparrow A$ to call an activity A by $\mathtt{a}$, $\mathtt{a}\diamondsuit_{g,p}$ a guarded, probabilistic, or conditional probabilistic decision in $\mathtt{a}$, $\mathtt{a}\blacklozenge$ a merge node, $\mathtt{a}|$ a join, $\mathtt{a}\|$ a fork, $\mathtt{a}\ddagger_p$ an interrupt region with a probability p, and $\rightarrowtail_g$ the activity edge with the guard g[1]. Definition 1 expresses formally SysML activity diagrams by assuming one initial node and one final node only for each diagram, and $Dist(N)$ denotes a convex probability distributions over the set of activity nodes N.

Definition 1. An SysML activity diagram is a tuple $A = \langle N,\ E, G,\ Grd,\ Prob \rangle$, where:

- N is a finite set of activity nodes such as a_i and a_f denote the initial and the final nodes, respectively; including $D = \{d_1, ..., d_l\}$ s a finite set of data objects,
- E is a finite set of activity edges,
- G is the set of guards,
- $Grd : E \mapsto G$ is a partial function that returns a guard for an edge, and
- $Prob : N \mapsto Dist(N)$ is a partial probabilistic function that assigns for each node a convex discrete probability distribution $\mu \in Dist(N)$ over its output transitions.

Based on Definition 1, two properties are proposed to shape the structure of SysML activity diagram.

Property 1 (Structure Constraint)**.** Let m represents the number of edges, and n is the number of nodes. Then, m > n.

Property 2 (Token Constraint)**.** In a SysML activity diagram S, let m represents the number of edges, and k is the number of tokens. So, k < m.

[1] g is a propositional logic formula that can be extended later.

During the execution, the structure of the activity diagram is kept unmodified and the only changes is the tokens locus. The NuAC syntax was inspired by this idea so that a NuAC term presents a static structure while tokens are the only dynamic elements. We can distinguish two main syntactic terms: marked and unmarked. A marked NuAC (New Activity Calculus) term corresponds to an activity diagram with tokens. An unmarked NuAC term corresponds to the static structure of the diagram. A marked term is typically used to denote a reachable state that is characterized by the set of tokens locations in a given term. We present in Figure 2.1 the Backus-Naur-Form of the calculus, called NuAC, that helps to formalize SysML activity diagrams. NuAC allows for multiplicity in join, merge, fork, and decision constructs by exploiting their commutativity and associativity properties. We denote by $A[N]$ to specify N as a sub term of A and by $|A|$ to denote a term A without tokens. For the call behavior case of $a \uparrow A'$, we denote $A[a \uparrow A']$ by $A \uparrow_a A'$ [13].

$$
\begin{array}{lcl}
A & ::= & \epsilon \quad | \quad \overline{l : \iota}^{n} \rightarrowtail N \\
N & ::= & \overline{N}^{n} \quad | \quad l : M(x,y) \rightarrowtail N \quad | \quad l : J(x,y) \rightarrowtail N \quad | \quad l : F(N,N) \quad | \quad \overline{l : a \uparrow A}^{n} \rightarrowtail N \\
 & & \quad | \quad l : D((p,g,N),(1-p,\neg g,N)) \quad | \quad l : \otimes \quad | \quad l : \odot \quad | \quad l
\end{array}
$$

Figure 2.1: Syntax of NuAC.

To support multiple tokens, we augment the "overbar" operator with an integer n such that $\overline{N}^{n}$ denotes a term marked with n tokens with the convention that $\overline{N}^{1} = \overline{N}$ and $\overline{N}^{0} = N$. Multiple tokens are needed when there are loops that encompass in their body a fork node [14]. Furthermore, we use a prefix label for each node to reference it and uniquely use it in the case of a backward flow connection (case of merge or join). Particularly, labels are useful for connecting multiple incoming flows towards merge and join nodes. Let L be a collection of labels ranged over by l_0, l_1, $\cdots$ and N be any node (except initial) in the activity diagram. We write $l : N$ to denote an l-labeled activity node N. It is important to note that nodes with multi-inputs (e.g. join and merge) are visited as many times as they have incoming edges. Thus, as a syntactic convention, we use either the NuAC term (i.e. $l : M(x,y) \rightarrowtail N$ for merge and $l : J(x,y) \rightarrowtail N$ for join) if the current node is visited for the first time or its corresponding label (i.e. l_x or l_y) if the same node is encountered later during the traversal process. Also, we denote by $D((g,N_1),(\neg g,N_2))$ or $D((p,N_1),(1-p,N_2))$ to express a decision without probabilities or guards, respectively. The execution of SysML activity diagrams is based on token's flow. To give a meaning to this execution, we use structural operational semantics to formally describe how the computation steps of NuAC atomic terms take place. The operational semantics of NuAC is based on the informally specified tokens-passing rules supported by SysML.

We define Σ as the set of non-empty actions labeling the transitions (i.e. the alphabet of NuAC, to be distinguished from action nodes in activity diagrams). An element $\alpha \in \Sigma$ is the label of the executing active node. Let Σ^{o} be $\Sigma \cup \{o\}$ where o denotes the empty action. Let p be a probability value such that $p \in]0,1[$. The general form of a transition is $A \xrightarrow{\alpha}_{p} A'$ and $A \xrightarrow{\alpha} A'$ in the case of a Dirac (non probabilistic) transition. The probability value specifies the likelihood of a given transition

to occur and it is denoted by $P(A, \alpha, A')$. Fig. 2.2 shows the operational semantic rules of NuAC. The semantics of SysML activity diagrams expressed using A as a result of the defined semantic rules can be described in terms of the PA (probabilistic automata) stipulated in Definition 2. In addition, we propose in Table 2.1 the NuAC axioms that are proved by using NuAC semantic rules.

Definition 2 (NuAC-PA)**.** A probabilistic automata of a NuAC term A is the tuple $M_A = (\overline{s},\ \mathrm{L},\ \mathrm{S},\ \Sigma^o,\ \delta)$, where:

- $\overline{s}$ is an initial state, such that $\mathrm{L}(\overline{s}) = \{\overline{l : \iota} \rightarrowtail N\}$,
- $\mathrm{L} : S \rightarrow 2^{[\![L]\!]}$ is a labeling function where: $[\![L]\!] : L \rightarrow \{\top, \bot\}$,
- S is a finite set of states reachable from $\overline{s}$, such that, $\mathrm{S} = \{s_{i:0\leq i\leq n} : \mathrm{L}(s_i) \in \{\overline{N}\}\}$,
- Σ^o is a finite set of actions corresponding to labels in A,
- $\delta : \mathrm{S} \times \Sigma^o \rightarrow\ Dist(S)$ is a partial probabilistic transition function such that, for each $s \in S$ and $\alpha \in \Sigma^o$ assigns a probabilistic distribution μ, where:
  - For $S' \subseteq\ S$ such that $S' = \{s_{i:0\leq i\leq n} : s \xrightarrow{\alpha}_{p_i} s_i\}$, each transition $s \xrightarrow{\alpha}_{p_i} s_i$ satisfies one NuAC semantic rule and $\mu(S') = \sum_{i=0}^{n} p_i = \sum_{i=0}^{n} \mu(s_i) = 1$.
  - For each transition $s \xrightarrow{\alpha}_1 s''$ satisfying a NuAC semantic rule, μ is defined such that $\mu(s'') = 1$.

INIT-1	$\overline{l : \iota} \rightarrowtail N \xrightarrow{l} l : \iota \rightarrowtail \overline{N}$
ACT-1	$\overline{l : a}^{m} \rightarrowtail N \xrightarrow{l} \overline{l : a}^{m-1} \rightarrowtail \overline{N} \quad \forall m > 0$
ACT-2	$\overline{\overline{l : a}^{m} \rightarrowtail N}^{n} \xrightarrow{l} \overline{\overline{l : a}^{m+1} \rightarrowtail N}^{n-1} \quad \forall m \geq 0, n > 0$
BH-1	$\dfrac{A = l' : \iota \rightarrowtail N' \quad \forall n > 0}{\overline{l : a \uparrow A}^{n} \rightarrowtail N \xrightarrow{l} \overline{l : a \uparrow \overline{l' : \iota} \rightarrowtail N'}^{n-1} \rightarrowtail N}$
BH-2	$\dfrac{A[\overline{l' : \odot}] \xrightarrow{l'} \lvert A\rvert \quad \forall n > 0}{\overline{l : a \uparrow A}^{n} \rightarrowtail N \xrightarrow{l'} \overline{l : a \uparrow A}^{n} \rightarrowtail \overline{N}}$
FORK-1	$\overline{l : F(N_1, N_2)}^{m} \xrightarrow{l} l : F(\overline{N_1}, \overline{N_2})^{m-1} \quad \forall m > 0$
PDEC-1	$\overline{l : D((p, g, N_1), (1-p, \neg g, N_2))}^{m} \xrightarrow{l}_p \overline{l : D((p, g, \overline{N_1}), (1-p, \neg g, N_2))}^{m-1} \quad \forall m > 0$
MERG-1	$A[\overline{l : M(x,y) \rightarrowtail N}^{n}, \overline{l_x}^{m}, \overline{l_y}^{k}] \xrightarrow{l_x} A[\overline{l : M(x,y) \rightarrowtail \overline{N}^{n}, \overline{l_x}^{m-1}, \overline{l_y}^{k}}] \quad \forall m > 0, k, n \geq 0$
MERG-2	$A[\overline{l : M(x,y) \rightarrowtail N}^{n}, \overline{l_x}^{m}, l_y] \xrightarrow{l_x} A[\overline{l : M(x,y) \rightarrowtail \overline{N}^{n}, \overline{l_x}^{m-1}, l_y}] \quad \forall m > 0, n \geq 0$
JOIN-1	$A[\overline{l : J(x,y) \rightarrowtail N}^{n}, \overline{l_x}^{m}, \overline{l_y}^{k}] \xrightarrow{l_x} A[\overline{l : J(x,y) \rightarrowtail \overline{N}^{n}, \overline{l_x}^{m-1}, \overline{l_y}^{k-1}}] \quad \forall m, k > 0, n \geq 0$
FLOWFINAL	$A[\overline{l : \bigotimes}] \xrightarrow{l} A[l : \bigotimes]$
FINAL	$A[\overline{l : \odot}] \xrightarrow{l} \lvert A\rvert$
ACTIVITY	$\dfrac{N \xrightarrow{\alpha}_p N'}{A[N] \xrightarrow{\alpha}_p A[N']}$

Figure 2.2: NuAC Operational Semantic Rules.

DA-1	$l\colon D((p,g,N_1),(1-p,\neg g,N_2)) = l\colon D((1-p,\neg g,N_2),(p,g,N_1))$
DA-2	$l\colon D((p,N_1),(1-p,l'\colon D((p',N_2),(1-p',N_3)))) = l\colon D((p+p'-p\times p', l'\colon D((\frac{p}{p+p'-p\times p'},N_1),(\frac{p'-p\times p'}{p+p'-p\times p'},N_2))),(1-p-p'+p\times p',N_3))$
DA-3	$l\colon D((p,g,N_1),(1-p,\neg g,l'\colon D((p',g',N_2),(1-p',\neg g',N_3)))) = l\colon D((p,g,N_1),(p'-p.p',\neg g\wedge g',N_2),((1-p)(1-p'),\neg g\wedge\neg g',N_3))$
FA-1	$l\colon F(N_1,N)_1 = N_1$
FA-2	$l\colon F(N_1,N_2) = l\colon F(N_2,N_1)$
FA-3	$l\colon F(N_1,l'\colon F(N_2,N_3)) = l\colon F(l'\colon F(N_1,N_2),N_3) = l\colon F(N_1,N_2,N_3)$
JA-1	$A[l\colon J(x,y)\rightarrowtail N', N\rightarrowtail l_x, N\rightarrowtail l_y] = A[N\rightarrowtail N']$
JA-2	$l\colon J(x,y)\rightarrowtail N = l\colon J(y,x)\rightarrowtail N$
JA-3	$A[l\colon J(x,x')\rightarrowtail N, l'\colon J(y,z)\rightarrowtail l_{x'}] = A[l\colon J(x,y,z)\rightarrowtail N]$
MA-1	$A[l\colon M(x,y)\rightarrowtail N', N\rightarrowtail l_x, N\rightarrowtail l_y] = A[N\rightarrowtail N']$
MA-2	$l\colon M(x,y)\rightarrowtail N = l\colon M(y,x)\rightarrowtail N$
MA-3	$A[l\colon M(x,x')\rightarrowtail N, l'\colon M(y,z)\rightarrowtail l_{x'}] = A[l\colon M(x,y,z)\rightarrowtail N]$
CA-1	$l\colon a\uparrow\epsilon = a$
CA-2	$A_1\uparrow_{a_1}(A_2\uparrow_{a_2}A_3) = (A_1\uparrow_{a_1}A_2)\uparrow_{a_2}A_3 = A_1\uparrow_{a_1}A_2\uparrow_{a_2}A_3$

Table 2.1: Axioms for NuAC.

After presenting probabilistic automata as a precise semantics to a system modeled in SysML, we will extend this obtained semantics to support cyber-physical systems with a social dimension, named *Socio-Cyber-Physical Systems* (SCPS).

2.2 Algebraic Specification

To model SCPS with a societal dimension [15], we defined a S as a tuple $\langle Phy, Obj, Act, Struc\rangle$, where Phy is the physical space made of rooms and locked/unlocked doors; Obj are the objects found in the space including containers holding physical as well as information content; Act are the actors (i.e. the people, malicious agent) and their interactions with and within the physical space; $Struc$ defines the structure of the physical space by telling what rooms are adjacent and which are connected by doors [7]. Formally, we describe the CPS tuple as follows:

A) Phy is a tuple $\langle L, D, key_D\rangle$, where

- L is a finite set of locations (with elements l, l', etc.).
- D is a finite set of doors (with elements d, d', etc.).
- $key_D\colon D\mapsto O$ is a partial function that returns the object (i.e. the key) that can lock/unlock a door. $Dom(key_D)$ is the set of doors that can be locked.

B) *Obj* is a tuple $\langle O, type_O, attr_O, key_O \rangle$, where

- O is a finite set of objects (with elements o, o', etc.).
- $type_O \colon O \to \{p, d\}$ returns the type of object, physical (p) or digital (d).
- $attr_O \colon O \to 2^{\{c,m,d,n\}}$ returns a set of attributes of an object, that is, container (c), movable (m), destroyable (d), and clonable (n).
- $key_O \colon O \mapsto O$ is a partial function that returns the object (i.e. the key) that can lock/unlock a physical object (resp. encrypt/decrypt a digital object). $Dom(key_O)$ are digital objects and containers that can be locked. Note that a password can lock a safe, and a physical key used to encrypt (lock) a file.

C) *Act* is a tuple $\langle A, I, \Sigma \rangle$, where

- A is a finite set of actors (with elements a, a', etc.).
- $I \notin A$ is the intruder. We use A_I as a shorthand for the set $A \cup \{I\}$.
- Σ is the set of basic actions that any agent can perform: moving from one location to another through a door, locking/encrypting, unlocking/decrypting, drop or pick objects from rooms or containers, destroy, clone information, exchange objects and information with other peers, or do nothing. Assuming that $l, l' \in L$, $d \in D$, $o, o' \in O$, $a \in A$, and $x \in L \cup O$, and $v \in D \cup O$, we have

$$\Sigma = \{MoveTo(d,l,l'), Lock(v,o), UnLock(v,o), Put(o,x), Get(o,x), \\ Destroy(o), Clone(o,o'), Give(o,a), Rec(o,a), Stop\}.$$

  The following actions as their names mean are related to a given actor where $MoveTo(d,l,l')$ leads to move from location l to l' through door d, $Lock(v,o)$ to lock v with o, $UnLock(v,o)$ to unlock v with o, $Put(o,x)$ to put o in x, $Get(o,x)$ to get o from x, $Destroy(o)$ to destroy o, $Clone(o,o')$ to clone o in o', $Give(o,a)$ to give o to a, $Rec(o,a)$ to receive o from a, and *Stop* means ending the actors' behavior.

- $bv_A \colon A \to L$ returns the expression that describes the behaviour of an actor. L is a language described by $B ::= \alpha \cdot (B +_x B)$ where $\alpha \in \Sigma$, '$\cdot$' and '$+_x$' are respectively the sequential and the decision operators (p for probabilistic where $p \in [0,1]$, g for deterministic when g is a propositional formula and $\top$ when non-deterministic). We assume a structural equivalence $=$ on expressions, defined as the smallest relation that satisfies the following equalities:
  - $B_1 + B_2 = B_2 + B_1$,
  - $B_1 + B_2 = B_1$ when $B_1 = B_2$,
  - $(B_1 + B_2) + B_3 = B_1 + (B_2 + B_3) = B_1 + B_2 + B_3$, and
  - $Stop.B = Stop$.

D) *Struc* is a multi-graph $\langle V, E, C \rangle$, representing how the entities are connected, where

- $V = \{b\} \cup L \cup A_I \cup O$, i.e. the vertices, is the set of all the entities plus b the root, which may be considered the name of physical space.
- $E \subseteq (\{b\} \times L) \cup (L \times (A_I \cup O)) \cup (A_I \times O) \cup (O \times O)$
- $\langle V, E \rangle$ is a tree rooted in b.
- $C \subseteq L \times D \times L$ is a set of edges labelled by doors representing the rooms' connection.

We represent the execution of actions in a `SCPS` by a labeled state transition system $\langle S, S_0, \Rightarrow \rangle$, where: S is the set of all possible `SCPS` states, $S_0 \in S$ is the initial state, and $\Rightarrow \subseteq (S \times \Gamma \times \mathbb{C} \times \mathbb{P} \times S)$ is the transition relation between states for a set of labels Γ, a probabilistic distribution $\mathbb{P}$, and a set of costs $\mathbb{C}$. It is the smallest relation that satisfies the transition rules. We express each transition by $s \xRightarrow{\ell,p,c} s'$, where the states are represented by the relevant part of the multi-graph. The edge labels and the nodes in the graph express elements of the state that relate to them. We display only those elements that express a condition for the occurrence of the transition or that change due to the transition.

Further, we define two auxiliary functions to keep track of the `SCPS`'s executed actions.

- $Hist_D : \vec{S} \times D \to 2^{\mathbb{B} \times A}$ returns a list of pairs, each pair saying whether the door has been locked (`true`) or unlocked (`false`) and the actors performing that action;
- $Hist_O : \vec{S} \times O \mapsto 2^{\mathbb{B} \times A}$ returns a list of pairs, each pair saying whether the object has been locked (`true`) or unlocked (`false`) and the actors performing that action.

To show the validity of the proposed semantics, we have considered Maroochy water services sewerage system that consists of 142 sewage pumping stations where each pumping station had a computerized system capable of receiving commands from a central control center (master station) and transmitting signals back to the center. The communication between pumping stations and the control center was through a private two-way radio system. Figure 2.3 shows the system architecture and the connection between its components. The control of the pumping stations can be through the main station or through one of the pumping stations access points as well as by agents when needed.

In the Maroochy Water Services presented in Figure 2.3, p_i is a pump, s_j^i a sensor that measures the status of the pump p_i, r_k a remote terminal unit (RTU), m a master station, a an access point, and h an agent who uses the access points with a laptop to manage the system. We used $bh(m)$ to describe the behaviour of the master station m where it can receive or send a command from/to RTU r_i. $bh(r_i)$ represents the behaviour of the RTU r_i where it can communicate with the master station m, another RTU r_j, a pump p_j, an access point a_j or a sensor s_j. $bh(p_i)$ is the behaviour of the pump, it could receive a command o_{r_j} from an RTU to change its state to start or stop running.

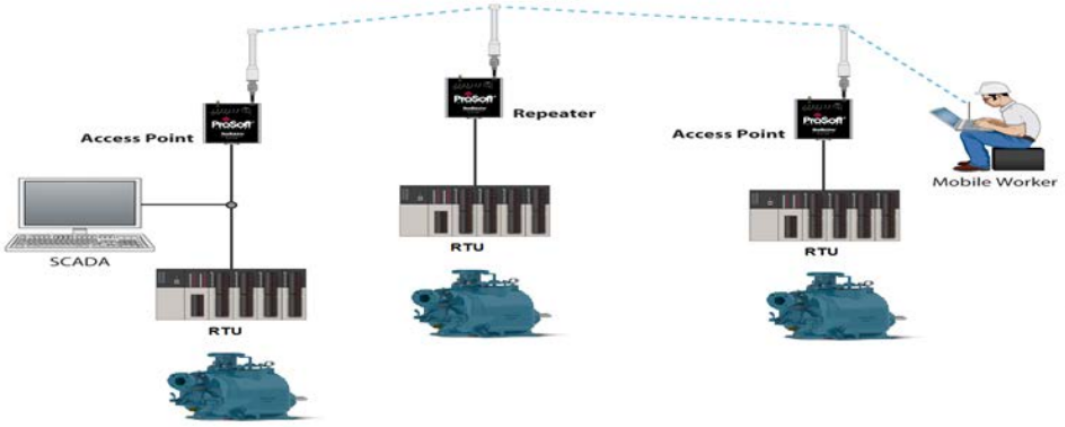


Figure 2.3: The Maroochy Water Services Sewerage System.

$bh(s_i)$ represents the behaviour of the sensor to receive the status of the pump and send it to the RTU. $bh(h_i)$ is the behaviour of an agent using the access point to receive and send commands. The behaviours of the entities showed in Figure 2.3 are expressed as follows.

- The behaviour of the master station m.
  $beh(m) = (Get([\![o_{r_i}]\!], r_i).Put(o_{i_m}, [\![o_{r_i}]\!]) + Get(r_i, [\![o_{i_m}]\!]) + Put(r_j, [\![o_{j_m}]\!])).Stop$

- The behaviours of the RTU r_i.
  $beh(r_i) = (Get([\![o_{m_i}]\!], m) + Get([\![o_{r_j}]\!], r_j) + Rec([\![o_{a_j}]\!], a_j) + Get([\![o_{p_j}]\!], p_j).$
  $Put(o_i, [\![o_i]\!]).Give(m, [\![o_{r_i}]\!]) + Give(r_j, [\![o_{r_j}]\!]) + Put(p_k, [\![o_{r_i}]\!])).Stop.$

- The behaviour of the pump p_i.
  $beh(p_i) = (Rec([\![o_{r_j}]\!], r_j).(Unlock(p_i, o_{p_i}) + lock(p_i, o_{p_i}))).Stop$

- The behaviour of the sensor S_i.
  $beh(s_i) = start.Rec([\![s_{p_j}]\!], p_j).Get(s_i, [\![s_{p_j}]\!]).Put(r_i, [\![s_{p_i}]\!]).Stop$

- The behaviour of the agent h_i.
  $beh(h_i) = (Rec([\![o_{a_j}]\!], a_j).Put(o_i, [\![o_{a_j}]\!]).(Put(a_{j'}, [\![o_i]\!] + Rec([\![o_{a_j}]\!], a_j)).Stop$

In the previously, we have introduced probabilistic decisions to refine the non-deterministic choices. However, since **SCPS** are large, it is evident the obtained probabilities is not easy at the modeling stage. Thus, introducing priority concept is an alternative that can help to precisely define a system as will be shown in the next section.

2.3 Behavioral Interaction Priority

To reorder actions and to orient tasks in CPS, we integrate priority mechanism to solve the non-determinism issue by relying o n Stochastic BIP. First, we recall Labelled Transition Systems (LTS) that constitutes the underlying semantics of the basic constructs of BIP components. The definition of LTS is portrayed as follows.

Definition 3 (Labelled Transition System)**.** A labelled transition system is a tuple $\langle Q, Act, \rightarrow, q_0 \rangle$, where:

- Q is a set of states,
- Act is a set of action names labeling the transitions,
- $\rightarrow \subset Q \times Act \times Q$ is a set of labelled transitions. For convenience, if $(q, a, q') \in \rightarrow$, intuitively we write $q \xrightarrow{a} q'$, and
- $q_0 \subseteq Q$ is a set of initial states.

Atomic components are elementary building blocks for modelling a system in BIP. They are described as labelled transition systems extended with *variables*. Transitions between states are labelled by *ports*. Eventually, a transition is associated with a guard g and an update function $Func(\vartheta)$, that are respectively, a propositional logic formula and a computation defined over local variables ϑ. $Eval(\vartheta)$ is a function that assigns values to variables ϑ. $[\![g(X)]\!]$ returns the Boolean evaluation of the condition g. An atomic component in BIP is formally defined as follows.

Definition 4 (Atomic Components)**.** An atomic component $B{=}\langle S, P, T, s_0, \vartheta \rangle$ is a labelled transition system extended with data, such that:

- $\langle S, P, T, s_0 \rangle$ is a labelled transition system, where: S is a set of states, P a set of communication ports, T is a set of transitions of the form (s, p, g, f, s') where s, $s' \in S$, $p \in P$, $g \in Eval(\vartheta)$ is a guard, and $f \in Func(\vartheta)$ is an update function on a subset of ϑ, and
- $s_0 \in S$ is the initial state, and
- $\vartheta = \{v_0, \ldots, v_n\}$ is a set of local variables.

Let $\mathbb{D}$ be a finite universal domain. Given a set of variables ϑ, we define valuations for variables as functions $X : \vartheta \rightarrow \mathbb{D}$ that associate each variable in ϑ with a value in $\mathbb{D}$. We denote the set of valuations of variables in ϑ as $\mathbb{D}^{\vartheta} = \{X_0, X_1, \ldots\}$. Also, we define a function $Var : P \rightarrow 2^{\vartheta}$ that associates each port in P with a set of variables in ϑ. Within the BIP atomic component, for a given valuation of variables, a transition can be executed if and only if its associated guard evaluates to *true*. When

several transitions are simultaneously enabled, a non-deterministic choice is performed to fire one of them and to execute its internal computation f. Definition 5 presents the semantics of BIP atomic components.

Definition 5 (Semantics of Atomic Components)**.** The atomic component $B = \langle S, P, T, s_0, \vartheta \rangle$ is an LTS $\langle Q, Act, \rightarrow, q_0 \rangle$, such that:

- $Q \subseteq S \times \mathbb{D}^{\vartheta}$,
- $Act \subseteq P$ is a pre-ordered set of transitions labels,
- $\rightarrow$ is a set of transitions of the form $((s;X),\ \mathrm{p},\ (s';X')))$ written $(s;X) \xrightarrow{p} (s';X')$ such that, $Var(p) \subseteq X$, $g(X)$ evaluates to true and X' is the new valuation for some $\tau = (s, p, g, f, s') \in T$, and
- $q_0 = (s_0; X_0) \subseteq Q$.

ENB-UPD 1: This axiom introduces the transition enabling if and only if the guard g is evaluated to true.

$$\frac{p{\in}P,\ var(p){\in}X,\ Dom(X){=}Dom(X'),X{\neq}X',\ [\![g(X)]\!]{=}\top}{(s;X)\xrightarrow{p}(s';X')}$$

ENB-UPD 2: This axiom solves the nondeterminism in the atomic component by following a policy. In BIP, we consider the priority as a strong policy that solves the non-deterministic choices.

$$\frac{p_1,p_2{\in}P,var(p_1){\in}X_1,var(p_2){\in}X_2,Prior(p_1){\geq}Prior(p_2),\ [\![g(X_1)]\!]{=}\top,[\![g(X_2)]\!]{=}\top}{(s_1{:}X_1)\xrightarrow{p_1}(s_1'{:}X_1')}$$

SYNC: This axiom highlights the synchronization between two atomic components on a port p such as $B_1|_p B_2$.

$$\frac{p{\in}P,\ var(p){\in}X,\ Dom(X){=}Dom(X'),\ B_1{=}(s_1{:}X_1)\xrightarrow{p}(s_1'{:}X_1'),B_2{=}(s_2{:}X_2)\xrightarrow{p},[\![g(X_1)]\!]{=}\top,[\![g(X_2)]\!]{=}\top}{((s_1,s_2){:}X_1{\cup}X_2)\xrightarrow{p}((s_1',s_2'){:}X_1'{\cup}X_2')}$$

PRB-UPD: This axiom presents the probabilistic update in one atomic component with a new probabilistic variable update $X'^{p}(v^p) = x^{v^p}$.

$$\frac{p{\in}P,\ var(p){\in}X,\ Dom(X^p){=}Dom(X^{p'}),\ Dom(X^d){=}Dom(X^{d'}),\ [\![g(X^d)]\!]{=}\top}{(s{:}X^p{\cup}X^d)\xrightarrow{p}(s'{:}X^{p'}{\cup}X^{d'})}$$

Figure 2.4: BIP Operational Semantics.

The stepwise behaviour of an atomic component B is described by the operational semantic rules in Figure 2.4. To deal with the system's interoperability, BIP formalism provides connection operators

that allow composing and coordinating atomic components called *glue*. The latter provides mechanisms for harmonizing and coordinating components behaviours, namely priorities and interactions. Indeed, to ensure a correct interaction, BIP *connectors* bind ports from different sub-components towards composition. For a model built from a set of components B_1, $B_2, \ldots, B_n$, where $B_i = \langle S_i, P_i, T_i, s_{0_i}, \vartheta_i \rangle$, we assume that their respective sets of ports and variables are pairwise disjoint, *i.e.* for any $i \neq j \in \{1, \ldots, n\}$, we require that $P_i \cap P_j = \emptyset$ and $\vartheta_i \cap \vartheta_j = \emptyset$. Thus, we define the set $P = \bigcup_{i=1}^{n} P_i$ of all ports in the model as well as the set $\vartheta = \bigcup_{i=1}^{n} \vartheta_i$ of all variables. An interaction is formally defined as follows.

Definition 6 (Interaction)**.** An interaction a is a triple $\langle P_a, g_a, f_a \rangle$ where: (1) $P_a \subseteq P$ is a set of ports related to the interaction a, (2) g_a is a set of guards, and (3) f_a is the data transfer function that takes the form of a sequence of functions expressed by $f_1 \odot \ldots \odot f_n$ and applied over data variables of the synchronized (i.e. interacting) components.

The result of the interaction is a composition of synchronized components obtained by using the component composition operator γ presented in Definition 7.

Definition 7 (Composition)**.** The composition of n atomic components denoted by $\gamma(B_1, \ldots, B_n)$ is a composite component $B = \langle S, P, T, s_0, \vartheta \rangle$, where:

- $S = S_1 \times \cdots \times S_n$ is a set of states tuples,
- P is a set of ports,
- T is the set of transitions where $\tau \in T$ takes the form $((s_1, \ldots, s_n), P_a, G_a, F_a, (s'_1 \ldots, s'_n))$ that results on synchronizing (i.e. written "$|_{P_a}$") a set of transitions $\{\tau_i = (s_i, prt_i, g_i, f_i, s'_i) \in T_i\}_{i \in I}$ where : $prt_i \in P_a$, $G_a = g_a \bigwedge_{i \in I} g_i$, $F_a = f_a \odot_{i \in I} f_i$ and $I \subseteq \{1, \cdots, n\}$,
- $s_0 = s_0^1 \times \ldots \times s_0^n$, and
- $\vartheta = \bigcup_{i=1}^{n} \vartheta_i$.

The semantics of the synchronized atomic components $B_1|_{P_a} B_2$ in LTS is described by the operational semantic rule "**SYNC**" in Figure 2.4. Also, the stepwise behaviour of the synchronized atomic components $B_1||B_2$ in LTS is described by the operational semantic rule "**SYNC**" in Figure 2.4. Nonetheless, the occurred synchronization if and only if ports are enabled at the same time. However, the states of components that do not participate in the interaction remain unchanged. Observe also that γ defines the set of all allowed interactions, it is also possible that an interaction P_a is never enabled in γ $(B_1, \ldots, B_n)$. So, we identify two cases of synchronization: *Rendez-vous* and *Broadcast* interactions. Figure 2.5 shows a components composition γ (S, R_1, R_2, R_3): a sender S and three receivers R_1, R_2 and R_3. The sender has the port S for sending messages and each receiver has a port r_i $i = \{1, 2, 3\}$ for receiving them. Table 2.2 specifies γ for two different coordination schemes.

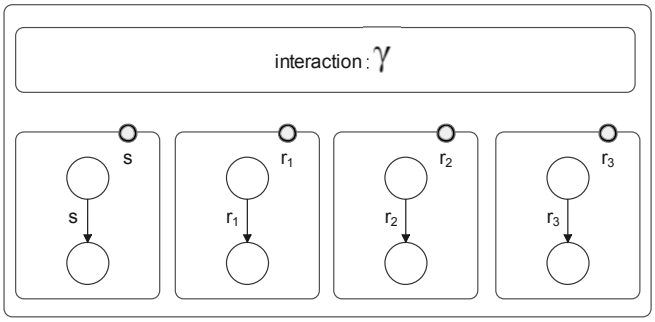


Figure 2.5: System with the Interaction of four Atomic Components.

Rendez-vous. This means strong synchronization between S and all R_i, specified by a single interaction involving all ports. This interaction can occur only if all components are in states enabling transitions labeled, respectively, by s, r_1, r_2, and r_3.

Broadcast. This allows all interactions involving any (possibly empty) subset of R_i. This is specified by the set of all interactions containing s. These interactions can occur only if S is in a state enabling s (i.e. s is the interaction initiator port). Each R_i participates in the interaction only if it is in a state enabling r_i.

	Set of interactions γ
Rendez-vous	$\{s\ r_1\ r_2\ r_3\}$
Broadcast	$\{s,\ s\ r_1,\ s\ r_2,\ s\ r_3,\ s\ r_1\ r_2,\ s\ r_1\ r_3,\ s\ r_2\ r_3,\ s\ r_1\ r_2\ r_3\}$

Table 2.2: Interaction Sets for Basic Coordination Schemes.

Stochastic BIP is an extension of BIP supporting probabilistic behaviour that will constitute the foundation upon which we will build our approach for verification and forecasting. Syntactically, we add stochastic behaviour at the level of atomic components by allowing the definition of probabilistic variables.

For the finite universal data domain $\mathbb{D}$, a probability distribution over $\mathbb{D}$ is a function $\mu : \mathbb{D} \to [0,1]$ such that, $\Sigma_{x_i \in \mathbb{D}} \mu(x_i) = 1$ for all $x_i \in \mathbb{D}$. Formally, a stochastic extension of an atomic component is stipulated as follows.

Definition 8 (Stochastic Atomic Components)**.** A stochastic atomic component is an atomic component extended with probabilistic variables $B^s = \langle S, P, T, s_0, \vartheta \rangle$, where:

- $S = \{s_1, \ldots, s_k\}$ is a set of states,
- P is a set of ports,
- $\vartheta = \vartheta^d \bigcup \vartheta^p$, with $\vartheta^d = \{v_1, \ldots, v_n\}$ the set of deterministic variables and $\vartheta^p = \{v_1^p, \ldots, v_m^p\}$ the set of probabilitic variables attached to a set of probability distributions $\mu_{\vartheta^p} = \{\mu_1, \ldots, \mu_m\}$.

- T is a set of transitions of the form $\tau = (s, prt, g, f, s')$ where $s, s' \in S$, $prt \in P$, g is a guard over $Eval(\vartheta)$, and f is a pair (f^d, f^p) where f^d is a deterministic update function on ϑ^d and f^p is a probabilistic update function on ϑ^p, and
- s_0 is the initial state.

We denote the set of valuations of a set of probabilistic variables ϑ^p as $\mathbb{D}^{\vartheta^p}$. ϑ^p is initially associated with a default valuation $X_0^p \in \mathbb{D}^{\vartheta^p}$. Each probabilistic variable $v^p \in \vartheta^p$ is attached to a probability distribution $\mu \in \mu_{\vartheta^p}$. We use small scripts x^{v^p} to denote valuations of single probabilistic variables $v^p \in \vartheta^p$. So, considering the definition of stochastic BIP, its semantics is amenable to an MDP (Markov Decision Process) as formally stated below.

Definition 9. The semantics of a stochastic atomic component $B^s = \langle S, P, T, s_0, \vartheta \rangle$ is an MDP $M = \langle S_{MDP}, Act, \delta, \Sigma, L \rangle$, where:

- $S_{MDP} \subseteq S \times \mathbb{D}^{\vartheta^p} \times \mathbb{D}^{\vartheta^d}$ is a set of states,
- $Act \subseteq P$ is a set of actions,
- $\delta : S^{MDP} \times Act \times S^{MDP} \to [0,1]$ is a probabilistic transition function that returns a probability p to produce a probabilistic transition of the form $s_{MDP} \xrightarrow{prt} s'_{MDP}$ where $s_{MDP} = (s; X^{v^p}; X^{v^d})$ and $s'_{MDP} = (s'; X'^{v^p}; X'^{v^d})$ are obtained by:
  - Unchanging the valuations of non-updated probabilistic variables on prt, i.e., $\forall v^p \notin f^p$, $X'^{v^p}(v^p) = X^{v^p}(v^p)$, and
  - Assigning new updates $x^{v^p} \in \mathbb{D}^{v^p}$ to the probabilistic variables v^p, i.e., $\forall v^p \in f^p$, $X'^{p}(v^p) = x^{v^p}$.
- Σ is the set of atomic propositions over ϑ, and
- $L : S_{MDP} \to \Sigma$ is the state labelling function.

The stepwise behavior of an atomic component B in MDP is described in Figure 2.4 by the operational semantic rule "**PRB-UPD**". The probability of a transition $s \xrightarrow{prt} s'$ is the product of the distributions μ^{v^p} of all $v^p \in f^p$ that is $\prod_{v^p \in f^p} \mu^{v^p}(x^{v^p})$.

Making a projection of the theoretical definition of stochastic BIP over the language notations is necessary for the understanding of the coming sections. A BIP code (i.e. a Program) is composed of a set of "m" components ($m > 0$) where the behaviour of each component is described as a set of statements that take the following form.

$$\boxed{on\ prt\ from\ s\ to\ s'\ provided(g)\ do\{\ f^d(v^d);\ x^{v^p} = X^p(v^p);\}}$$

"*prt*" is the port labelling the transition preceded by a keyword "**on**" and forces components to synchronise and execute actions simultaneously in a lock-step fashion. The current state "*s*" is preceded by a keyword "**from**" and the next state "*s'* " is preceded by the keyword "**to**". The transition is enabled when the Boolean expression g evaluates to true using the Boolean condition $Eval(\vartheta)$ within the construct "**provided** ()". X^p is an update function on the probabilistic variable ϑ^p whereas f^d is a deterministic update function on ϑ^d.

As presented, we considered priority concept to solve non-determinism and make our system more realistic. Thus, one of the most systems properties that can be guaranteed are safety, and especially for automotive systems. As a cause study, we use BIP to model a trains transportation system. Figure 2.6 depicts the global graphical architecture of the complete trains movement and control system. Its equivalent BIP model is obtained from an automatic translation and online accessible [16]. A connector is required to transfer the computed velocity and acceleration to the target train. For instance, when `train1` synchronizes with the controller it should receive the already sent information in addition to those of `train2`. Thus, a conditional transfer is applied over the connector interaction on the id of the controller and trains as modelled in Listing 2.1 (lines 5-14).

Listing 2.1: Control to Train Parameters Transfer

```
1  connector type transfertTrainVCMandACM (Port_Type_Acc_Speed
       train1, Port_Type_Acc_Speed train2, Port_Type_Acc_Speed
       aatc)
2  define aatc train1' train2'
3  on train1 down {train1.fault = FAULTY ; }
4  on train2 down {train2.fault = FAULTY; }
5  on aatc train1 provided (aatc.id==train1.id)
6       down {
7       train1.speed=aatc.speed;
8       train1.acc=aatc.acc;
9       }
10 on aatc train2 provided (aatc.id==train2.id)
11      down {
12      train2.speed=aatc.speed;
13      train2.acc=aatc.acc;
14      }
15 end
```

2.4 Conclusion

The presented semantics have been dedicated to large scale systems with different components that support non deterministic and probabilistic choices. We have proposed a semantic proper to System Modeling Language (SysML), Social Cyber Physical Systems, and Behavioral Interaction

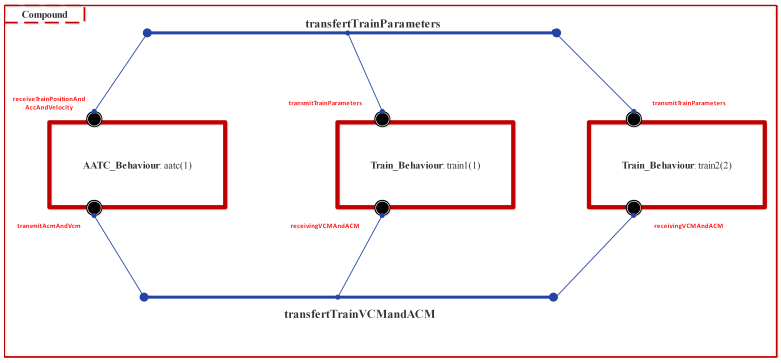


Figure 2.6: Graphical BIP Representation of Automatic Train Transport Systems .

Priority (BIP) modeling paradigm with the aim to facilitate the implementation ang automatically generate the code of **SCPS** after constructing their correct designs. We also showed how those contributions advancing the state-of-the-arts where the proposed formalism have been validated on standard modeling languages and experimented on model checking and statistical analysis tools.

Chapter 3

Security and Reliability for SCPS

Content

Ensuring security and reliability of **SCPS** is important as their functional correctness. Many languages and approaches are proposed in the literature to specify such complexity. One of the most security related issues to **SCPS** are: security specification, access controls, and hardening. We have developed different specification languages and techniques to specify smart attacks [17], security [18] and reliability [10] requirements, and access control policies [7]. Also, to complement security, it is important to show how the system is reliable before and after a reinforcement. Hence, we have developed mechanisms to reinforce security through policies [7, 8], security protocols [19], and blockchain [20]. The content of this chapter synthesise the results obtained from the following listed contributions.

1. **Ouchani, Samir**, "A security policy hardening framework for socio-cyber-physical systems," *Journal of Systems Architecture*, vol. 119, p. 102259, 2021.

2. **Ouchani, Samir** and G. Lenzini, "Generating attacks in sysml activity diagrams by detecting attack surfaces," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–13, 2015.

3. Dahmane Walid, Miloud, **Ouchani, Samir**, and H. Bouarfa, "Towards a reliable smart city through formal verification and network analysis," *Computer Communications*, vol. 180, pp. 171–187, 2021.

4. Fahem, Zerrouki, **Samir, Ouchani**, and H. Bouarfa, "Towards a foundation of a mutual authentication protocol for a robust and resilient puf-based communication network," *Procedia Computer Science*, vol. 191, pp. 215–222, 2021. The 18th International Conference on Mobile Systems and Pervasive Computing (MobiSPC).

5. Dahmane Walid, Miloud, **Ouchani, Samir**, and H. Bouarfa, "Guaranteeing information integrity through blockchains for smart cities," in *International Conference on Model and Data Engineering*, pp. 199–212, Springer, 2021.

6. Zerrouki, Fahem, **Ouchani, Samir**, and H. Bouarfa, "A generation and recovery framework for silicon pufs based cryptographic key," in *International Conference on Model and Data Engineering*, pp. 121–137, Springer, 2021.

7. K. Abd El-Aziz, **Samir, Ouchani**, T. Zahir, and D. Khalil, "Assessing the Severity of Smart Attacks in Industrial Cyber Physical Systems," *Transactions on Cyber Physical Systems*, 2020.

8. Baouya, Abdelhakim, O. A. Mohamed, D. Bennouar, and **Ouchani, Samir**, "Safety analysis of train control system based on model-driven design methodology," *Computers in Industry*, vol. 105, pp. 1–16, 2019.

3.1 Security Statements

In our paradigm, a *security policy* constraints the systetraces by restricting the actors' behaviour. When a policy is enforced, it limits what can happen. For instance, what agents can do on objects and files is determined by access control systems; how people behave is determined by accepted moral or social rules, or by regulations that people follow in fear of punishment. In modelling policies for `SCPS` case, we abstract the real reason of the enforcement and focus on the effect of that policies over the `SCPS` traces. Unlike security policies concept, a security requirement is a property that we would like to hold on the constrained `SCPS` [15]. The requirement must hold despite specific threats, coming from an adversary or from people acting dishonestly, that is, breaking the rules and the regulations assumed on the `SCPS` [7]. First, we discuss a way that specify and differentiate a policy and requirement. Then, we define a set of templates expressing the policies.

3.1.1 Security Policies vs. Security Requirements

We express policies and requirements using the language of *security statements* given in Definition 10 based on the next operator and bounded until of Linear Temporal Logic (LTL).

Definition 10 (security statement)**.** A *security statement* denoted by ϕ is an expression of the language generated by the following grammar, whose rules are written in Backus-Naur form:

$$
\begin{array}{rcl}
\phi & ::= & \psi_{SP} \mid \psi_{PL} \mid \psi_{TL} \\
\psi_{SP} & ::= & d \in conn(l,l') \mid o \in key_D(d) \mid (x,a) \in Hist_D(d) \mid \\
 & & y \in attr_O(o) \mid z \in type_O(o) \mid loc_O(o) = l \mid o \in key_O(o') \mid o \in cont_O(o') \mid \\
 & & (x,a) \in Hist_O(o) \mid loc_A(a) = l \mid o \in cont_A(a) \\
\psi_{PL} & ::= & \top \mid \neg\phi \mid \phi \wedge \phi \\
\psi_{TL} & ::= & \bigcirc\phi \mid \phi \mathrm{U}^{\leq k} \mid \phi \mathrm{U} \phi \mid \diamond\phi \mid \Box\phi
\end{array}
$$

The boolean operators $\wedge$ and $\neg$ give the full power of propositional logic; operators $\bigcirc$ and $\mathrm{U}^{\leq k}$ are sufficient to derive the other linear temporal operators.

$$
\varphi_1 \mathrm{U} \varphi_2 \stackrel{def}{=} \varphi_1 \ \mathrm{U}^{\leq \infty} \ \varphi_2 \qquad \diamond\varphi \stackrel{def}{=} \texttt{true}\mathrm{U}\varphi \qquad \Box\varphi \stackrel{def}{=} \neg\diamond\neg\varphi
$$

From the non-terminal symbol φ_{SP} we derive the *state propositions*, which are propositions over a `SCPS`'s state. Their informal meaning can be evinced from the name of the statement. So, for instance, $o \in key_D(d)$, where o ranges over the objects O, evaluates true if and only if o is the key that opens door d. The formal semantics is given in Table 3.1. It defines $[\![\cdot]\!]_S$ the function that returns a φ's truth value given a particular `SCPS`'s state $S \in S$. All the items (set of nodes, labels, edges) in Table 3.1 must be intended as those defined in S.

$$
\begin{array}{rl}
[\![d \in conn(l,l')]\!]_S & \text{iff } (l,d,l') \in C \\
[\![(x,a) \in Hist_D(o)]\!]_S & \text{iff } (x,a) \in Hist_D(o) \\
[\![(x,a) \in Hist_O(o)]\!]_S & \text{iff } (x,a) \in Hist_O(o) \\
[\![y \in attr_O(o)]\!]_S & \text{iff } y \in attr_O(o) \\
[\![z \in type_O(o)]\!]_S & \text{iff } z \in type_O(o)
\end{array}
\qquad
\begin{array}{rl}
[\![loc_O(o) = l]\!]_S & \text{iff } (l,o) \in (E)^+ \\
[\![loc_A(a) = l]\!]_S & \text{iff } (l,a) \in E \\
[\![o \in key_D(d)]\!]_S & \text{iff } o = key_D(d) \\
[\![o \in key_O(o')]\!]_S & \text{iff } o = key_O(o') \\
[\![o \in cont_O(o')]\!]_S & \text{iff } (o',o) \in (E)^+ \\
[\![o \in cont_A(a)]\!]_S & \text{iff } (a,o) \in (E)^+
\end{array}
$$

Table 3.1: Interpretation of the state formulas give an `SCPS`'s state S. Here E^+ is the transitive closure of E

.

The semantics of φ is the standard semantics of LTL formula. It is the set $Words(\varphi) = \{\rho \in 2^{\varphi_{SP}} : \rho \models \varphi\}$ of all ω-words (i.e. infinite words) that satisfy φ, where the satisfaction relation $\models \subset 2^{\varphi_{SP}} \times \mathrm{LTL}_\varphi$ is the smallest relation satisfying the following properties (here, if $\rho = s_1 \cdots s_2 \ldots$, $\rho[i] = s_i \cdots s_{i+1} \ldots$):

- $\rho \models \top$
- $\rho \models \varphi_{SP}$ iff $[\![\varphi_{SP}]\!]_{\rho[0]}$
- $\rho \models \neg\varphi$ iff $\rho \not\models \varphi$
- $\rho \models \varphi_1 \wedge \varphi_2$ iff $\rho \models \varphi_1$ and $\rho \models \varphi_2$
- $\rho \models \bigcirc\varphi$ iff $\rho[1\ldots] = S_1 \ldots \models \varphi$
- $\rho \models \varphi_1 U \varphi_2$ iff $\exists\, j \geq 0, \forall 0 \leq i < j$: $\rho[j\cdots] \models \varphi_2$ and $\rho[i\cdots] \models \varphi_1$

We model a policy that can bound the SCPS behaviour (e.g. agent a should never enter in location l) which explicitly affects the SCPS's semantics (e.g. there will be no state where a is in location l). This raises the question about whether the real policy, that we are modeling, is actually capable to cause on the system (e.g. is it really true that imposing that a cannot enter in l, implies that there will be no state where a is in l?). This may depend on the policy, but we will chose to model policies in such a way that the answer of this question is positive (e.g. yes it is true that imposing an actor a to not enter in l implies that there will be no state where a is in l). In contrast, the requirement describes functional and non-functional guidelines to ensure the behavioral correctness of the SCPS model within the policy. Further, a requirement needs to be satisfied in order to ensure security and the correct behavior of an SCPS model.

In our SCPS context, we consider a security policy as a safety property or the negation of a liveness property. We assert that a security requirement any security statement that the SCPS model must satisfy. We denote by π a security policy and by ϕ a security requirement.

Definition 11 (Security Policy)**.** A *security policy* is a security statement that expresses either a safety property of the form $\Box\neg\varphi_{SP}$ or a negation of a liveness property written as $\neg\Box(\varphi_{SP} \rightarrow \Diamond\varphi_{SP})$.

Definition 12 (Security Requirement)**.** A *security requirement* is any security statement ϕ.

3.1.2 Security Policies Templates.

We propose now a templates of security policies that can be generated for any SCPS. Policies 1.a-1.f are about the intruder and statements 2.a-2.c are about integrity, confidentiality, and authentication.

1. The statements based on the intruder capabilities are stated in six policies:

   (a) Locking an unlocked door without possessing the key.
   $\forall a \in A: \Box\neg((key_D(d) \notin cont_A(a) \wedge d \in conn(l,l') \wedge \neg locked_D(d) \wedge loc_A(a) = l) \rightarrow (\bigcirc\neg(locked_D(d) \wedge lockedby_D(d) = a)))$.

   (b) Unlocking a locked door without having a key.
   $\forall a \in A: \Box\neg((key_D(d) \notin cont_A(a) \wedge d \in conn(l,l') \wedge locked_D(d) \wedge loc_A(a) = l) \rightarrow (\bigcirc\neg(\neg locked_D(d) \wedge lockedby_D(d) = a)))$.

   (c) Locking an unlocked container without having a key.
   $\forall a \in A: \Box\neg((key_O(o) \notin cont_A(a) \wedge \neg locked_O(o) \wedge loc_A(a) = loc_O(o)) \rightarrow$
   $(\bigcirc\neg(locked_O(o) \wedge lockby_O(o) = a)))$.

(d) Unlocking a locked container without having a key.
$\forall a \in A:\ \Box\neg((key_O(o) \notin cont_A(a) \wedge\ locked_O(o)\ \wedge\ loc_A(a) = loc_O(o))\ \rightarrow$
$(\bigcirc\neg(\neg locked_O(o) \wedge lockby_O(o) = a)))$.

(e) Stealing an object.
$\forall a, a' \in A:\ \Box\neg((o \in cont_A(a) \wedge loc_A(a) = loc_A(a')) \rightarrow (\bigcirc\neg(o \in cont_A(a'))))$.

(f) Slipping an object.
$\forall a, a' \in A:\ \Box\neg((o \in cont_A(a') \wedge loc_A(a) = loc_A(a)) \rightarrow (\bigcirc\neg(o \in cont_A(a))))$.

2. The statements based on the actor's behaviour: for a given object o and an actor a, let L_o and L_a be the sets of the authorized locations for o and a, respectively, O_a be the set of objects authorized to be possessed by a; and O_o the set of container authorized to contain o, we categorize these statements as follows.

(a) Integrity of locations and objects from modification; even if the actions that modify locations and objects do not exist in the set of actions of our language, but this capability can be a power of the attacker.

i. Integrity of locations: For any state, a given location stays the same as in the next state. $\forall l \in L : \Box(l \in L \rightarrow \bigcirc(l \in L))$.

ii. Integrity of objects: for any state, an object is the same as in the next state. $\forall o \in O ::$ $\Box(o \in O \rightarrow \bigcirc(o \in O))$.

iii. Integrity of doors. $\forall l, l' \in L : \Box((d \in conn(l, l')) = \bigcirc(d))$.

iv. Changing the key of a container. $\forall o \in O : \neg\Box(k \in key_O(o) \mathrm{U}\ k \notin key_O(o))$.

v. Changing the key of a door. $\forall d \in D : \neg\Box(k \in key_D(d) \mathrm{U}\ k \notin key_D(d))$.

(b) Confidentiality of holding objects by specific actors and/or in specific locations.

i. A confidential object should not be destroyed. In all states, a specific destroyable object always exists.
$\Box(((o \in O) \wedge (\{d\} \subseteq attr_O(o)))$

ii. An object should not be destroyed during a period of time 'T'.
$\Box(((o \in O) \wedge (\{d, f\} \subseteq attr_O(o)))\mathrm{U}^{\leq T}((o \notin O)))$

iii. A confidential and destroyable object should not be destroyed only if a condition c is satisfied.
$\Box(((o \in O) \wedge (\{d, f\} \subseteq attr_O(o)))\mathrm{U}\ (c\ \wedge\ o \notin O))$

(c) Authorization; ensures the authorized locations for objects and actors, and ensures the possession of objects by actors.

i. An object o should be only in a specific location of L_o.
$\forall o \in O:\ \Box(loc_O(o) \in L_o),\ \Box(loc_O(o) \notin L \backslash L_o)$.

ii. An object o should exist only in specific containers of O_o.
$\forall o \in O: \ \Box\neg(o \notin O \backslash O_o)$.

iii. An actor a is allowed to be only on locations L_a.
$\forall a \in A: \ \Box(loc_A(a) \in L_a), \ \Box(loc_A(a) \notin L \backslash L_a)$.

iv. An actor a is allowed to possess only objects in O_a.
$\forall a \in A: \ \Box(cont_A(a) \subseteq O_a)$.

v. An actor a is allowed to possess objects in O_a only in specific locations L_a.
$\forall a \in A: \ \Box(cont_A(a) \subseteq O_a) \rightarrow (loc_A(a) \in \ l)$.

3.2 Attacker model

Dolev-Yao attacker model [17] is the most powerful one that can access and manipulate arbitrarily all the network traffic. This attacker model is usually employed for the identification attacks (e.g, Web Application). It can intercept messages and analyze them if he possesses the corresponding keys for decryption. Also, it can generate messages from his knowledge, and send them as any honest or impersonate agent. However, this attacker model suffers from the state explosion and affects only the network layer as a main-in-the-middle. Our proposed attacker model is an improved version using the reinforcement learning and the multi-criteria analysis, where the attacker can perform all type of attacks (i.e. network, physical, software, and social engineering). In addition, we avoid the state explosion problem by relying on the reinforcement learning based on the multi-criteria analysis that makes the decision more deterministic and smart.

We describe potential attacks proper to `ICPS` (Industrial CPS) components and their interactions. As shown in Figure 3.1, the `ICPS` attacker model has two components: 1) *Data* contains knowledge, personality and skills, and 2) the *Analyzer* that is an inference based engine generating attacks according to data, the received inputs, and its skills and inference levels. The attacker model could be one from the internal employer who works inside or outside (e.g. disgruntled employees or a social engineering victims), a malicious software, or any node with powerful capabilities and techniques.

The `ICPS` attacker depicted in Figure 3.1 takes as input the channel $Chan$, the physical access Phy, and the human interaction Hum. $Chan$ means that the attacker intercepts the message between the `ICPS` components. Phy means that the attacker has physical access (e.g. open a door). Hum means that the attacker interacts or communicates with an employ from inside.

3.2.1 Data

The data ω of the attacker is a tuple $\langle K, P, S \rangle$, where K represents its knowledge, P is the personality of the attacker, and S is the set of skills.

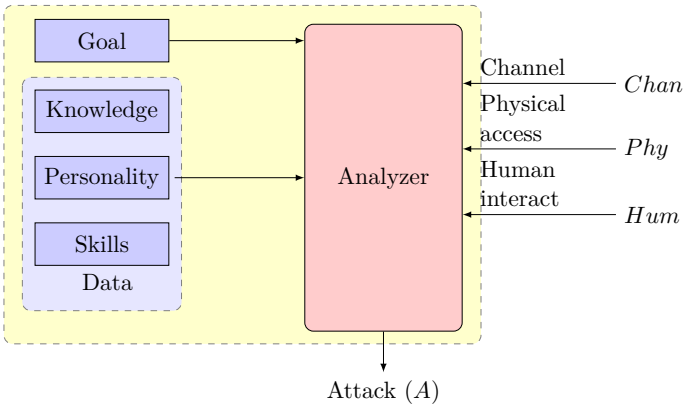


Figure 3.1: ICPS attacker model.

Knowledge

This is the core component of the attacker model. It contains the system model (M), a secure information (Sec), and algorithms of control or security techniques (Alg). Formally, the knowledge of the attacker (K) is a tuple, $K = \langle M, Sec, Alg \rangle$. Consequently, the knowledge in data can change according to the type or the level of attackers.

Personality

To fulfill the personality requirements, we rely on the well know theory from psychology called *the big five personality traits*, and also known as the five-factor model (FFM). Each factor represents a type of personality (Openness to experience, Conscientiousness, Extra-version, Agreeableness, Neuroticism) and the highest factor from the five above is the personality of the person. In our work, we are interested in the last factor Neuroticism, because the people who have a high level in this factor have emotions like: anger, anxiety, depression, and vulnerability. Formally, the personalty is a vector Per, where each element is an emotion e in the state i.

$$Per = \begin{vmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{vmatrix} \text{ where } \forall i \in [1,n],\ e_i = \begin{vmatrix} 1 & positive \\ 0 & absence \\ -1 & negative \end{vmatrix}$$

Goal

An attack is a set of actions or small attacks where each action has a goal, and formally a goal G is a set $G = \{g_1, g_2, \ldots, g_n\}$. An attacker must achieve all goals where g_i is an expression specifying a state of nodes.

3.2.2 Analyzer

Since a given attack A is a set of actions, the choice of this set depends on the set of the selected criteria. We use Markov Decision Process (MDP) and a multi-criteria analysis to model the decision of the attacker as well as the reinforcement learning to select the optimal sequence of actions to maximize the attacker assets. Algorithm 1 describes how the attacker could choose an action based on its goal while minimizing the cost.

Algorithm 1 Generation attack actions.

Input: M_{mdp} //The system behaviour with a graph structure.
Output: π //The attack sequence of actions.
$\forall s \in S:\ U(s) = 0;$
$\lambda = 0.5;$
$update = 1;$
while $update > \epsilon$ **do** //Update the cost U to achieve s.
  for $(s \in S)$ **do**
    $U(s) = R(s) + \lambda \max_{a \in A(s)} \sum_{s' \in S} P(s', s, a) \times U(s');$ //Measure the cost of a state s.
    $update = \sum_{s' \in S} |U(s) - U(s')|;$
  end for
end while
$\pi(s) = arg\max_{a} \sum_{s' \in S} P(s', s, a) \times U(s');$ //Select the path maximizing the reward.

Algorithm 1 takes as input a Markov Decision Process model M_{mdp} to produce the set of actions π. M_{mdp} is a tuple $M_{mdp} = (S, A, s_0, R, P, \gamma)$, where:

- S is a set of finite states $s_2, s_2, etc.$
- A is a set of finite actions $a_1, a_2, etc.$
- s_0 is the initial state.
- $R(s)$ is a reward function that returns the utility for each state s.
- $P(s, a, s')$ the probability of being in the state s' after executing the action a from state s.
- γ is a discount factor $0 < \gamma < 1$.

To achieve a state s_i in the model representing a sub-goal g_i, the attacker should execute an action a_i according to a sequence of decisions based on data. $R(s)$ is calculated with respect to the Weight Sum Method as follows

$$R(s) = \sum_{j=1}^{m} w_j c_{ij},\ i = 1, 2, ..., n$$

whre c is a set of criterion to select an action and w denotes the relative weight of importance of c. The probability of a transition $P(s, a, s')$ depends on the value of $R(s')$ and the sum of all the $R(s')$ for the successors of the state s. $P(s, a, s') = \dfrac{R(s')}{\sum_{\forall s' \in succ(s)} R(s')}$

The selection of a transition depends of the utility of each state and the accumulated reward. In our case, the attacker will choose the actions to earn more rewards. To deal with attacks, as the case in industrial CPS, we provided a blockchain based access management that is based on priorities in mining, as will be detailed in the next section.

3.3 Blockchain Access Management

We have developed a blockchain proper to smart cities [20] where each part can connect to the other as illustrated in Figure 3.2 by containing IoT devices, manager, advisers and miners. The advisor is the first installed component in the blockchain network to grand managers adresses and record a new manager as illustrated in Figure 3.3.

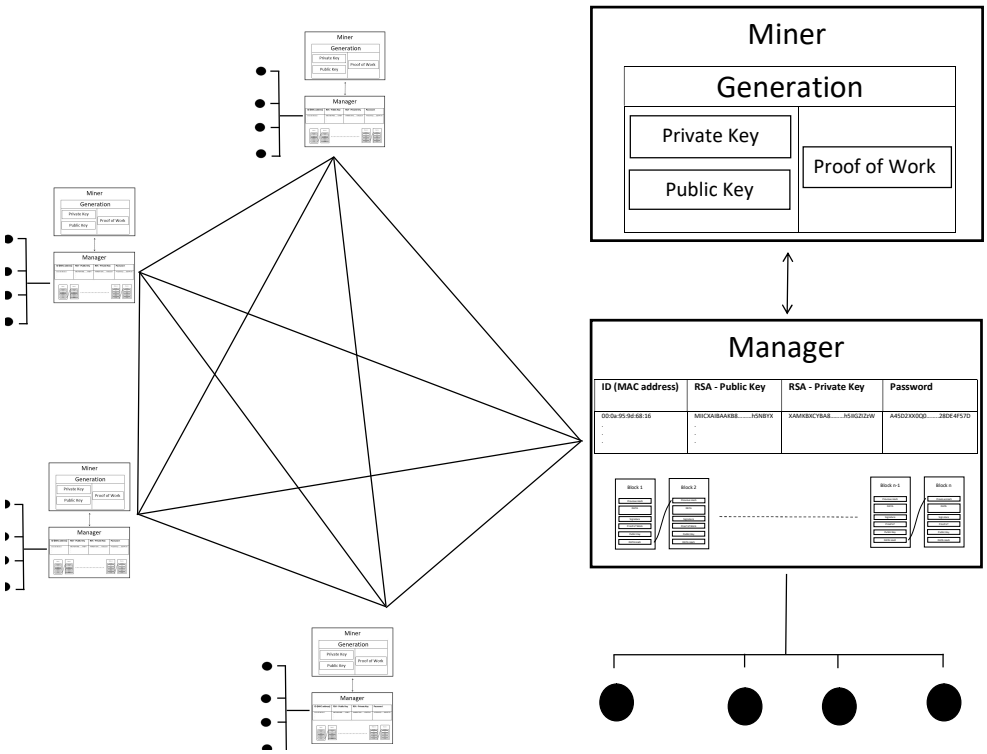


Figure 3.2: Blockchain network.

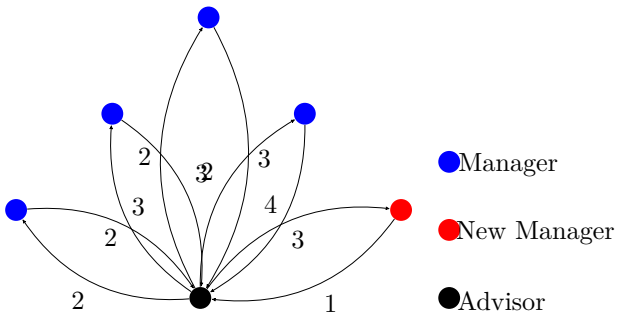


Figure 3.3: Record New Manager

3.3.1 The dominance of fraud

Our proposed blockchain network depends on the dominant blockchain in the network. This concept may impose a weakness when the malicious users dominate the majority of the network (or more then 50% malicious). The attackers will participate a single malicious blockchain that serves their interests, causing a loss of network reliability. This risk occurs in two cases, when spreading malicious managers in the network, or when hacking managers. We propose **V**alidation through **C**onfidence - **A**lgorithm (VCA) 2 against this type of threat. It should be installed in all the managers of network before the attacks occur. The algorithm consists of the "confidences criteria variable" given to the managers. This variable is increasing if the verification processes are correct. The variable will be converted into ranks in order to classify it with the others manager.

VCA defines the interesting ranks and their ratios of managers which will test the new block, the chosen random managers will be recorded in a table. It passes the new block to the managers in order to count the managers which accept this block. If the ratio of the acceptation decision equals the ratio of rejecting decision, the algorithm will extract a new random population. VCA can decrease the confidence criteria of the managers which have minorities decision instead of deleting them from the network (to reduce the punishment). We do not install the VCA in the advisor due to: make the blockchain network decentralised, maintaining the transparency concept where the users are the owners, and the damage of the advisor by an attack cause the absence of the VCA service [20, 8].

3.3.2 Verification time

The continual increase in the number of nodes and blockchain size in the network causes a decrease in the speed of sharing blocks. Equation 3.1 calculates the time spent ($\mathbb{T}$) in order to check the blockchain of the manager created by others managers, where $\mathbb{N}$ is the number of managers, $\mathbb{S}$ is the size of the blockchain and $\mathbb{B}$ is the time spent to check one block.

$$\mathbb{T} = \mathbb{N} \times \mathbb{S} \times \mathbb{B} \tag{3.1}$$

To solve this problem, we determine the ratio of managers checkers τ_1 and the ratio of the blocks which will be verified τ_2. We note that the blocks that must be verified are the last blocks of the blockchains for quick access to them. The new time $\top$ will be calculated through Equation 3.2.

$$\top = \tau_1 \times \tau_2 \times \mathbb{T} \tag{3.2}$$

The curve presented in Figure 3.4 displays the time spent $\top$ in terms of the ratios τ_1 and τ_2, considering that the original time $\mathbb{T}$ is one minute. Note that the smaller the two ratios τ_1 and τ_2, the smaller the time it takes to verify the blockchain $\top$. However, too much decrease in the two ratios may cause the network to lose its robustness.

3.3.3 Access Control Blockchain

In the access control process presented in Figure 3.5, we consider five types of blockchains, the difference among them is at the type of the *Data*.

Algorithm 2 Validation through the Confidence Algorithm (VCA)

```
1: procedure VCA(NewBlock)
2:     Define the rank and its ratio                ▷ e.g: Rank_A ← 60% ; Rank_B ← 20%.
3:     do
4:         for i ← 1, N do                          ▷ N: The number of the nodes chosen.
5:             Confidence [i] ← Random(Rank)        ▷ Fill the confidence table by random nodes.
6:         end for
7:         for i ← 1, N do
8:             Res ← Block_accept(Confidence [i], NewBlock)
9:             if Res == True then
10:                decision_Accept ++               ▷ Count the nodes which accept the new block.
11:            end if
12:        end for
13:    while decision_Accept_ratio == 50
14:    if decision_Accept_ratio > 50 then
15:        Add_Block(NewBlock)
16:        Increase_Node()   ▷ Increase the confidence criteria of the nodes which accept the new
       block.
17:        Delete_Node()                            ▷ Delete nodes which reject the new block.
18:    else
19:        Increase_Node()   ▷ Increase the confidence criteria of the nodes which reject the new
       block.
20:        Delete_Node()                            ▷ Delete nodes which accept the new block.
21:    end if
22: end procedure
```

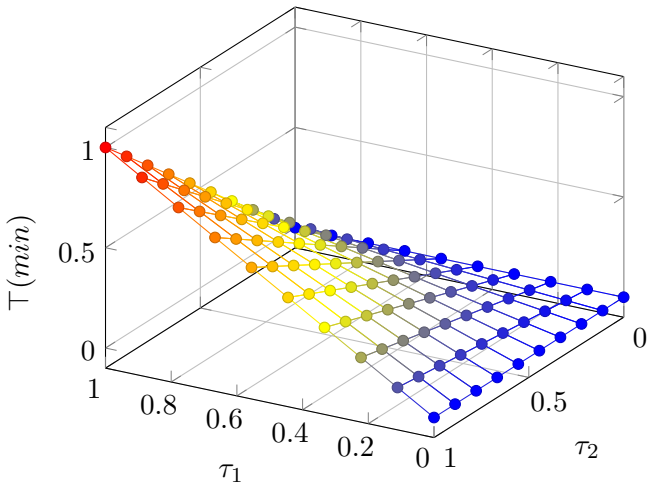


Figure 3.4: Time taken in terms of the ratios τ_1 and τ_2.

- **Subject-Blockchain:** Every new subject which joins the network is registered in the blockchain by creating his own block, that contains its ID, position, value and date of creation.
- **Object-Blockchain:** Every object should be valued for network subject through determining in each block the following parameters: ID object, ID subject, value and date of creation.
- **Task-Blockchain:** The tasks that the users perform are coded and valued by the parameters: ID Task, value and date of creation.
- **Threshold-Blockchain:** Every task applied to an object must equal or exceed a threshold defined on this type of blockchain, that is characterised by: ID Task, ID Object, value and date created.
- **Smart contract:** We have two types of policies that are integrated in this type of blockchain by specifying the parameters: ID Policy, Rules (or policies) and date of creation.

With the high increase of IoT devices, it is better to take the MAC address as an identification for the hardware components, due to the large number of identities it provides 2^{48}. It is described by the following formulas: An *Object* is a the tuple $\langle ID_{Ob}, Func_{Ob}, Numb_{Ob}, Comm_{Ob} \rangle$, where:

- ID_{Ob}: is a set of unique identifications suggested to be a $MAC\,addresses$,
- $Func_{Ob} = \{Send(inf),\ Receive(),\ Grant\_Access(Subject, Task),\ Record(inf)\}$ is a set of operations.
- $Numb_{Ob}$: The $Network$ can be equipped with an unlimited number of $Objects$;
- $Comm_{Ob}$: allows the $Object$ to link with the following components, where: $Comm_{Ob} = \{ Conn(Advisor[1]),\ Conn(Subject[n]) \mid n \in \mathbb{N}\}$.

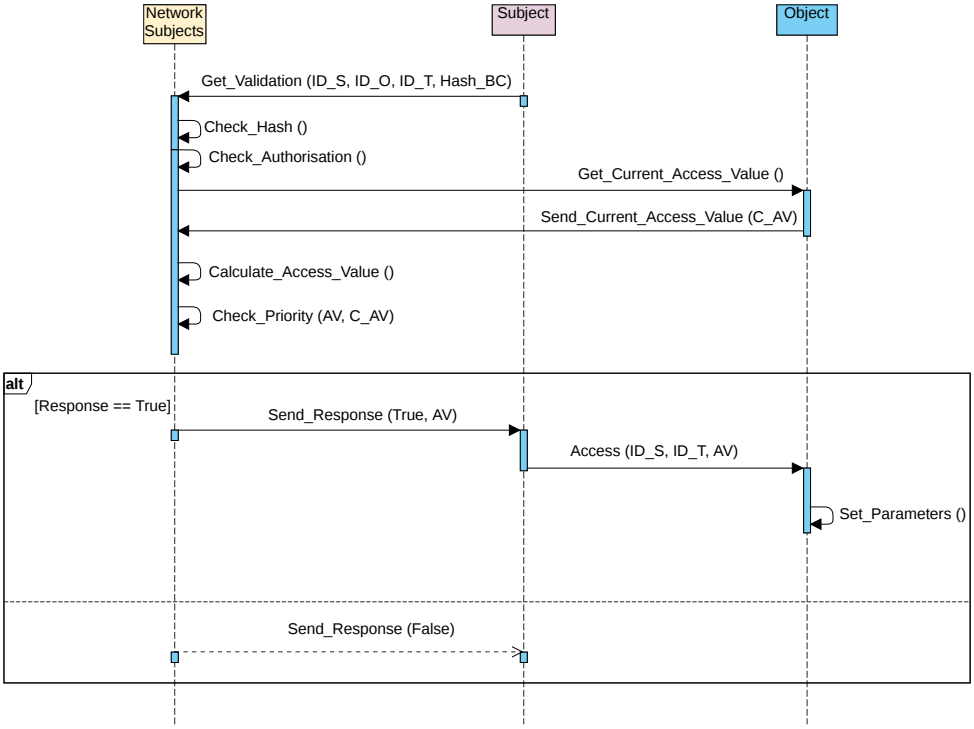


Figure 3.5: Access Control Processes.

For miners, first is to generate a hash of the peer (data + proof of work), the hash must respect a predefined condition, where the more difficult is the condition, then the more difficult is the process. The difficulty of condition depends of the sensibility of the data. The second function is to create the private and the public RSA keys for each subject. A Miner is the tuple $\langle ID_{Mi}, Func_{Mi}, Numb_{Mi}, Comm_{Mi} \rangle$; where:

- $Numb_{Mi}$: The $Network$ has at least one $Miner$; where: $if\ Network \neq \varnothing \Rightarrow \exists\ Miner \mid Numb_{Mi} \in \mathbb{N} \wedge\ Numb_{Mi} \neq 0$.
- $Comm_{Mi}$: it allows to connect with the types; where: $Comm_{Mi} = \{Conn(Advisor[1])\}$.

Since sensors are constrained resources and weak in term of security, it is a need tho provide a security protocol to deal with kind of components. We found that physical unclonable functions are one of the prominent alternatives to make our **SCPS** more secure.

3.4 PUF based Security

Due to the manufacturing process of IoT, SPUF (Silicon Physical Unclonable Functions) [22] is classified as a source of randomness and it can be used to generate a cryptography key without storing

the key's information for future use. This advantage is guaranteed since the key can be generated on demand. Nevertheless, one of the biggest challenges with SPUF is to guarantee the stability of the key in each regeneration phase, which is caused by the environmental variables. To achieve the stability (zero error rate) of the PUF response, it usually needs to be post-processed. The latter is generally composed of two phases: *enrollment* and *reconstruction* (see Figure 3.6). Our work is based on IoT randomness to develop more secure protocols [23].

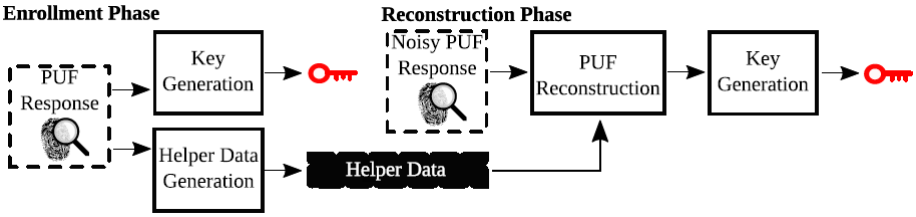


Figure 3.6: Enrollment and reconstruction phases.

3.4.1 PUF-based authentication protocol

To strength security in IoT based systems, we proposed mutual authentication protocol between an IoT device equipped with an arbiter PUF and the server. Physical unclonable function, a hashing function, and a fuzzy extractor are employed in our scheme. The protocol consists of three modules; 1) the enrollment phase, 2) the authentication phase, and 3) session key establishment. The used symbols and cryptographic functions are defined in Table 3.2.

Symbols	Definitions
ID_A	The identity of a IoT device A
Reg_{req}	Registration request
$Auth_{req}$	Authentication request
$C_{A,i}$	The i^{th} challenge of the device A
h(.)	One-way hash function
PUF_A	The PUF of a device A
$R_{A,i}$	A response of the challenge $C_{A,i}$
$R'_{A,i}$	A noisy response of $C_{A,i}$
Gen(.)	Generation procedure of Fuzzy Extractor
Rep(.,.)	Reproduction procedure of Fuzzy Extractor
$K_{A,i}$	Extracted key from $R_{A,i}$
$P_{A,i}$	Helper data of $R_{A,i}$
TS	Timestamps
$\|\|$	Concatenation symbol

Table 3.2: Authentication protocol's symbols.

3.4.2 Enrollment Phase

Contracting the most existing PUF-based authentication protocols that store a considerable number of CRPs wherein the authentication process the server challenged the IoT device with one randomly selected pair, and at the end, it will be deleted from the server database. In our proposed mechanism, the server stores only one pair. It minimizes the security threat due to the confidentiality of the stored data and resources of the server database [24]. In this phase, when a new IoT device needs to be added as a member of the trusted network, it goes first with the server through the enrollment phase. This phase is executed in a trusted and secure environment. Figure 3.7 describes the main steps of this phase to be performed by the $IoT-Device-A$ and the $Trusted-server$ are as follows.

- $IoT-Device-A$ sends its identity Id_A in plain text to the $Trusted-server$ with a registration request Reg_{req}.
- $Trusted-server$ generates randomly a challenge $C_{A,i}$, and sent it to $IoT-Device-A$ within ID_A.
- The $IoT-Device-A$ inputs this challenge into its arbiter PUF component to output the corresponding response $R_{A,i}$=$PUF_A(C_{A,i})$. Then, $IoT-Device_A$ sends to the server $ID_A, C_{A,i}, R_{A,i}$.
- Using the generation procedure of fuzzy extractor Gen, the $Trusted-server$ extracts the secret key $K_{A,i}$ and the public helper data $P_{A,i}$, ($K_{A,i}, P_{A,i}$)=Gen($R_{A,i}$). Then, the server computes the hash of the device identity and the secret key and stores h(ID_A), $C_{A,i}$, $h(K_{A,i})$, $P_{A,i}$ on its local secure database.
- In the end, $Trusted-server$ informs $IoT-Device-A$ about the end of the registration process.

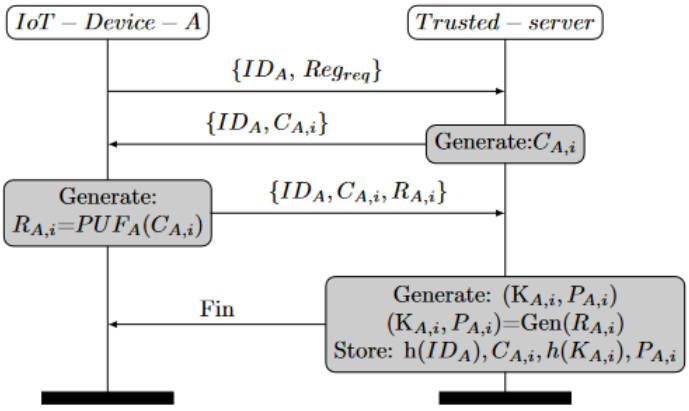


Figure 3.7: Enrollment Phase.

3.4.3 IoT device authentication Phase

Our proposed mutual authentication scheme is based on deploying a silicon PUF, especially a strong arbiter PUF in the IoT device. The idea behind using the PUF is that the IoT device uses the challenge-response pair of the PUF as the fingerprint and uses it to prove its identity with the server. The device and the server achieve mutual authentication since only those who know about the generated secret key for a given challenge in the enrollment phase and stored on the trusted server. In this proposed protocol, the server stores only one pair of CRPs (Challenge Response Pairs) to avoid attacks on the server. One of the most vital points of our protocol is that the IoT device does not store any secret and public information, which avoids physical attacks [19].

In the authentication phase to check the device's identity, PUF-based authentication protocols mainly compare the stored response in the enrollment phase to the new generated one for the same given challenge. Unfortunately, generating the same response for the same challenge in different environments and conditions such as voltage and temperature makes the response noisy/hazarded compared to the original one. This step is processed differently in our case, so to generate the secret key and store it safely on the server for the authentication process, the error correction technique has been adopted to eliminate the noise and ensures the comparison operation. More precisely, the proposed protocol takes into consideration the noise elimination process using the fuzzy extractor.

The authentication process between the IoT device (IoT-Device-A) and the server (trusted-server) is running as follows. First, in Step (1), the IoT-device-A generates a timestamp TS_1 and calculates a hash value of its Identity (ID_A) and $h(ID_A,TS_1)$. Then, the IoT device sends the hash of its identifier $h(ID_A)$, the authentication request $Auth_{req}$ and $h(ID_A,TS_1)$ message to the server.

In Step (2), upon receiving the message from the IoT device, first, the server checks the existing of the received $h(ID_A)$ in its database. If the finding fails, the server rejects the authentication request. Otherwise, the server verifies the received message integrity by calculating $h(ID_A,TS_1)$ message and matches both the calculated hash messages within the received one. If the matching fails, the server rejects the authentication request. Otherwise, the server makes sure that the message was not corrupted or tampered during the transmission phase. To calculate the message $h(C_{A,i},P_{A,i},h(K_{A,i}),TS_2)$, the trusted server retrieves $C_{A,i},P_{A,i},h(K_{A,i})$ that belongs to the ID_A from its database to its memory, and it generates a timestamp TS_2. Finally, the server sends $h(ID_A)$, the stored challenge $C_{A,i}$, the helper data correspondent to this challenge $P_{A,i}$, TS_2 and the message $h(C_{A,i},P_{A,i},h(K_{A,i}),TS_2)$ to the IoT device A.

In Step (3), once the IoT device receives the server response, the IoT device uses its arbiter PUF to generate the response of the received challenge $R'_{A,i}$=$PUF_A(C_{A,i})$. The generated response is considered noisy. Then, it reproduces the secret key $K_{A,i}$ = from the noisy response $R'_{A,i}$ using fuzzy extractor reproduction process $K_{A,i} = Rep(R'_{A,i},P_{A,i})$. In order to verify the integrity of the received message from the server, it calculates $h(C_{A,i},P_{A,i},h(K_{A,i}),TS_2)$, and compares the received and the

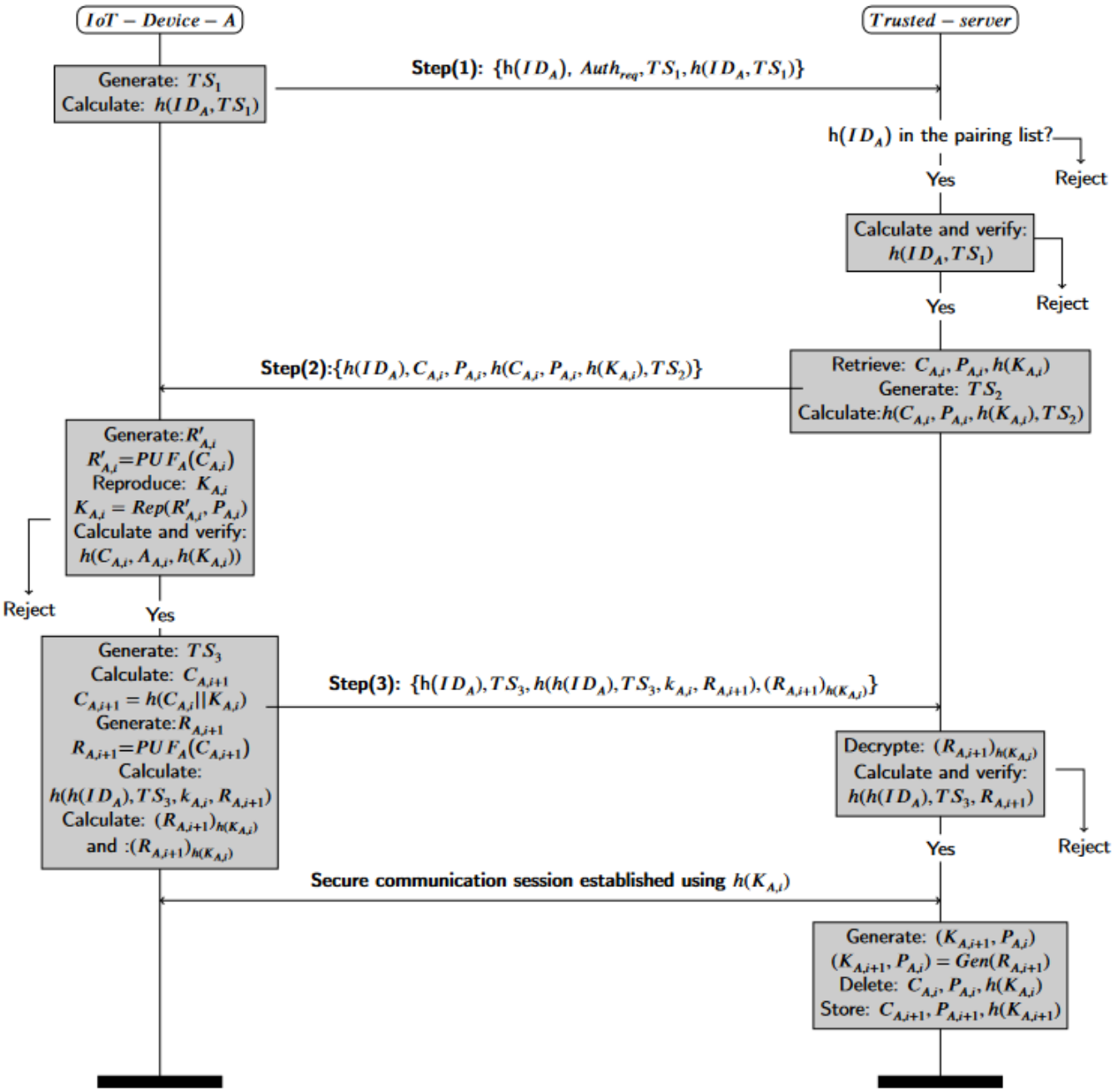


Figure 3.8: Proposed mutual authentication Protocol

calculated hash messages. This provides the first factor for authenticating the server, where the IoT device A verifies the authenticity of the server based on the success of matching the hash messages. This can be successful because the server is the only device in the network that knows the secret key $K_{A,i} =$ generated from the response $R_{A,i} =$ of $C_{A,i} =$. If the comparison fails, the connection is rejected by the IoT device. Otherwise, the IoT device generates a timestamp TS_3, computes a new challenge $C_{A,i+1} = h(C_{A,i}||K_{A,i})$, and generates the corresponding response of the new computed challenge $R_{A,i+1}$=$PUF_A(C_{A,i+1}$. Finally, the device calculates $h(h(ID_A),TS_3,k_{A,i},R_{A,i+1})$, encrypts the new response with the reproduced secret key $(R_{A,i+1})_{h(K_{A,i})}$, and sends back the calculated messages with ID_A and TS_3.

Upon receiving the message from the IoT device, the server decrypts $(R_{A,i+1})_{h(K_{A,i})}$ using the stored secret key $h(K_{A,i})$, and verifies the integrity of the received data by calculating h(h(ID_A),TS_3, $k_{A,i}$,$R_{A,i+1}$). Then, it compares the received and the calculated hash. If the matching fails, the server terminates the connection. Else, the server verifies the authenticity of the IoT device as a successful matching of these two hash messages. It means that the IoT is authenticated, and the server communicates with the right IoT device. At this step, the server and the IoT device can use the secret, and the exchanged key $h(K_{A,i}$ as session key to secure communication during the current session.

After a successful authentication of the IoT device, the server generates a new secret key and the corresponding helper data from the new generated response $R_{A,i+1}$, using fuzzy extractor generated procedure, $(K_{A,i+1}, P_{A,i}) = Gen(R_{A,i+1})$, and calculates the new challenge $C_{A,i+1} = h(C_{A,i}||K_{A,i})$. Finally, the server replaces the used information $C_{A,i}, P_{A,i}, h(K_{A,i})$ by the new one $C_{A,i+1}, P_{A,i+1}, h(K_{A,i+1})$ to be used in a future authentication process.

From our perspective, it is not only to make a system more secure but the question is how to secure a system while ensuring its continuous reliability, as will be presented in the next section.

3.5 Reliability

Reliability, Availability, and Maintainability (RAM) [12, 25] have been indispensable in the design phase of critical infrastructure in order to achieve minimum failures and thus to plan maintainability strategies, optimize reliability [26], and maximize availability [13]. In addition, the deployment is a post-production process that consists of software components [27] for use and keeping them operational and up-to-date. Thus, the deployment process consists of automatically explores the states' space of possible allocations of software blocks to hardware blocks according to functional constraints and returns the set of near-optimal candidates. In our work, software deployment is driven by the reliability of the system services that must be maximized. Service refers to the flow of actions and data at the software level. We assume that software failures due to programming imperfections are unlikely influencing the deployment process, then we may abstract away from the hardware-independent

software reliability. Also, the software and hardware architecture is constant during the deployment.

Our approach assumes that processors are failing silent [28], [29]; It means that the processor detects all errors and switch immediately to a passive state, which leads to an exception at the software level. When failures occur during the execution of software blocks, it impacts the reliability of our system. With a fixed and deterministic scheduling strategy, any failure that happens on the processor leads to service execution failure. To evaluate a single deployment, we use some specification metrics for the software and the platform architecture to capture the system reliability. The constraints established in this section correspond to the attributes of components in the SysML specification. For instance, the processor capacity corresponds to the $<< HwProcessor >>$ featured with MIPS (million instructions per second) as MARTE annotation.

The set of execution components represented by processor are denoted by U $=\{ u_1, u_2, ..., u_m \}$, where $m \in \mathbb{N}$ and the parameters of hardware architecture are given as follows:

- Processor speed, ps : U $\rightarrow \mathbb{N}$; the instruction-processing capacity of the hardware component in MIPS.
- Processor failure rate, λ : U $\rightarrow \mathbb{N}$; the rate parameters of the Poisson distribution that characterizes the probability of a single processor failure.

The set of software components that must be allocated to processor are denoted by C $=\{ c_1, c_2, ..., c_n \}$, where $n \in \mathbb{N}$ and the parameters of software components are given as follows:

- Component workload, wl : C $\rightarrow \mathbb{N}$; computational requirement for component expressed in MI (millions instructions)

Let D= { d| d:C $\rightarrow$ U } denotes the set of all functions assigning software components to hardware processors, and we write a single deployment alternative $\mathrm{d}_j = \{ (c_1, u_{j1}), (c_2, u_{j2}), ..., (c_n, u_{jm}) \} \subseteq D$ as $\mathrm{d}_j = \{ u_{j1}, u_{j2}, ..., u_{jm} \}$. The quality metric of the approach depends on the reliability of physical hardware (Processors and buses). The reliability of the physical elements is computed according to the failures of execution elements (Processors). The reliability of single element i [28] can be modeled with exponential distribution as follows where λ is the failure rate of the element and T is the time elapsed in it.

$$\mathrm{R_i} = \mathrm{e}^{-\lambda.\mathrm{T}} \tag{3.3}$$

In this model, failure rates of execution elements are obtained from the hardware architecture parameters, and the time taken for the execution relies on the component workload and processing speed [28]. In time triggered-architecture [29], processing depends on the fixed scheduling algorithm. Thus, the requests are queued until their time slot arrives. Based on the software workload and processing speed of the processors, the scheduling length can be calculated as follows:

$$sl(u_i) = \sum_{c \in d^{-1}(u_i)} \frac{wl(c)}{ps(u_i)} \tag{3.4}$$

The reliability of individual software block are computed as :

$$\mathrm{R_i} = \mathrm{e}^{-\lambda(d(c_i))sl(d(c_i))/2} \tag{3.5}$$

The result R_i refers to the reliability of each allocation of software component that allows populating our activity diagram with probabilities or could be parameters for the PRISM model checker. We note that the probabilities on the activity diagram are unset (i.e., variables). When the reliability of individual software blocks is known, the full reliability is computed using the PRISM model checker applied over the parameterizable activity diagram, and we refer to the full reliability as R_{d_j}.

Having the reliability measure R_{d_j} for each candidate under deployment $d \in D$, the reliability R of the best candidate is calculates as :

$$\mathrm{R} = Max\{R_{d_0}, ..., R_{d_m}\}, \quad m \in \mathbb{N} \tag{3.6}$$

To investigate the reliability impact on the deployment, we model an train control system (AATC) composed of a set of execution elements (i.e. Processors) in BIP language; the BIP code is extended with the computed probabilities. We use the defined parameters above to enhance our model.

In the case of our analysis, the safety degree corresponds to a certain level of reliability in which the system is in an operating state at any time. It is assumed that communications between the train control (AATC) and trains do not generate anomalies that affect the system's reliability.

Considering the formula of Equation 3.5, we refine the model by adding a reachable failing state when a transition at each phase in AATC processors of the train system does not occur. So, Each computation phase is susceptible to fail; then, the model is refined as in Figure 3.9. The resulting model is around 463 lines of code long, and online access from [30]. The refined model highlights a new state `S5` that represents the case where the system is not available. However, we have to model a correct computation by checking the availability of the system; for instance, to model a transition `S1` $\rightarrow$ `S2` we check the value of the probabilistic variable x_R that takes 0 or 1, whereas `Reliable` is set to 1. The `fault` variable refers to the failure data reception between AATC processors, whereas `FAULTY` is set to 0:

```
on computeVCMCivilSpeed from S1 to S2 provided (!(fault==FAULTY) && (x_R==Reliable)) do{ }
```

In the other case, when the AATC fails (i.e. One of the processors fails), a transition in a dashed blue color (Figure 3.9) is represented textually as follow.

```
on computeVCMCivilSpeedFail from S1 to S5 provided (!(x_R==Reliable)) do { }
```

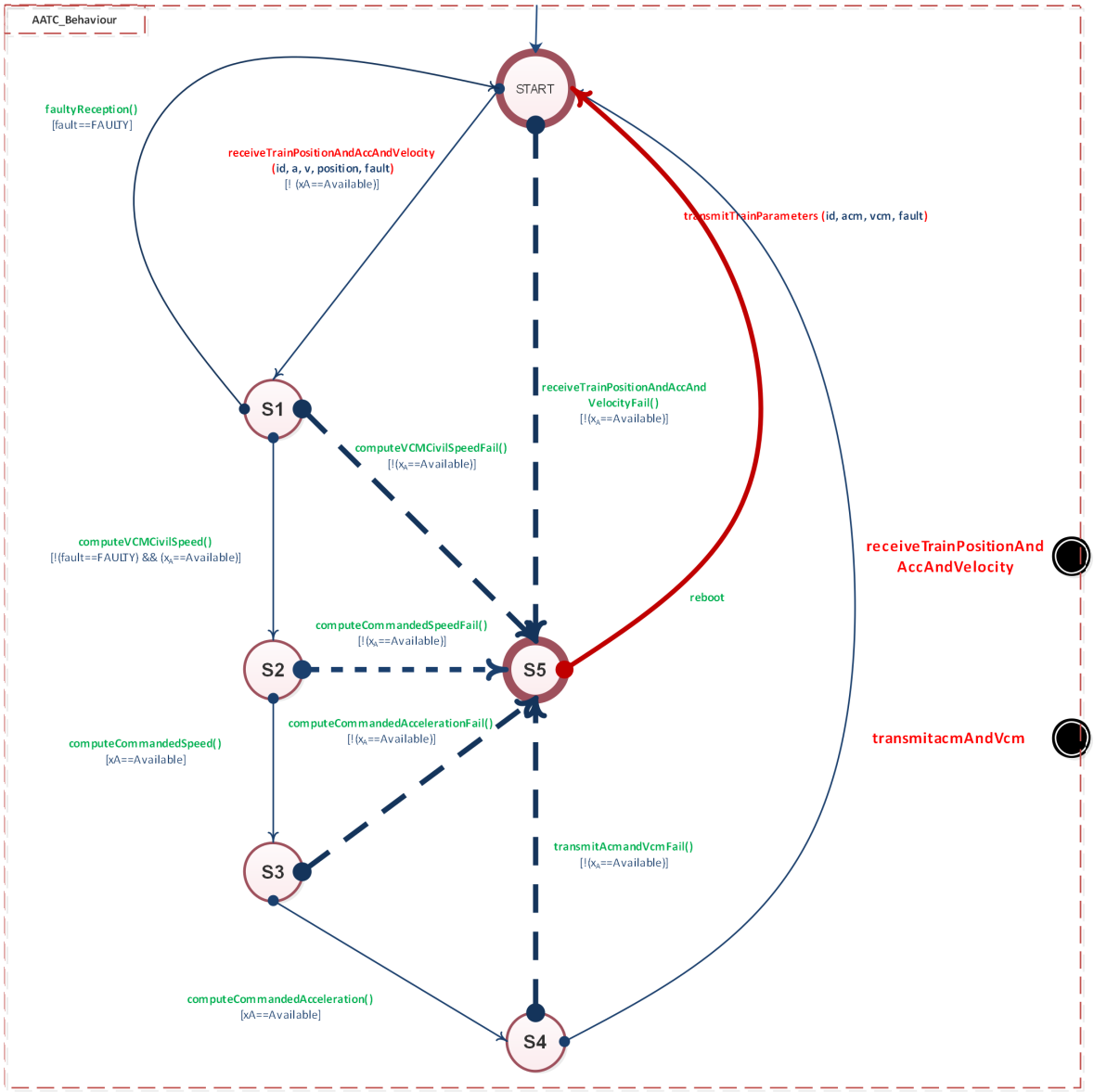


Figure 3.9: Refined BIP model of AATC system with reliability parameters.

Meanwhile, the unavailability of AATC implies the system rebooting (transition portrayed in Figure 3.9 with red color). Textually, the transition is represented as follows.

```
on reboot from S5 to START do { }
```

3.6 Conclusion

In this chapter, we have specified and presented many techniques to precisely express reliability and security properties as well as presenting sound and adequate semantics. In addition, we have presented smart attacks with different strategies that have been successfully applied on real use cases, especially industrial cyber-physical systems. Further, we developed a Blockchain and PUF-based security protocol to manage efficiently access controls of large and constrained components. Furthermore, we have studied reliability in real industrial use cases, especially trains and automotive systems sensitive to reliability changes that affect directly the security and the functionality of **SCPS**.

Chapter 4

Analyzing SCPS

Content

Analyzing and checking vulnerabilities and weaknesses of such system is important as its modeling and security but challenging due to the complexity, heterogeneity, and the size of **SCPS**. Thus, to automate the analysis, we rely more on the model checking approach [6, 17]. Then, we enhance the verification process by introducing abstract [31] and composition [32] techniques for probabilistic checking and also by using stochastic simulation and probabilistic verification [9, 8]. Further, we correct such errors by hardening automatically a system by proper policies [7]. This chapter tackles these issues by summarizing the following contributions:

1. Abdelhakim Baouya, O. A. Mohamed, **Samir Ouchani**, and D. Bennouar, "Reliability-driven Automotive Software Deployment based on a Parametrizable Probabilistic Model Checking," *Expert Systems with Applications*, p. 114572, 2021.

2. K. Abd El-Aziz, **Samir, Ouchani**, T. Zahir, and D. Khalil, "Assessing the Severity of Smart Attacks in Industrial Cyber Physical Systems," *Transactions on Cyber Physical Systems*, 2020.

3. Dahmane Walid, Miloud, **Ouchani, Samir**, and H. Bouarfa, "Towards a reliable smart city through formal verification and network analysis," *Computer Communications*, vol. 180, pp. 171–187, 2021.

4. Baouya, Abdelhakim, O. A. Mohamed, D. Bennouar, and **Ouchani, Samir**, "Safety analysis of train control system based on model-driven design methodology," *Computers in Industry*, vol. 105, pp. 1–16, 2019.

5. Baouya, Abdelhakim, S. Chehida, **Ouchani, Samir**, S. Bensalem, and M. Bozga, "Generation and verification of learned stochastic automata using k-nn and statistical model checking," *Applied Intelligence*, Nov 2021.

6. **Ouchani, Samir**, "Towards a fractionation-based verification: application on sysml activity diagrams," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pp. 2032–2039, 2019.

7. **Ouchani, Samir**, "Towards a call behavior-based compositional verification framework for sysml activity diagrams," in *International Colloquium on Theoretical Aspects of Computing*, pp. 216–234, Springer, Cham, 2019.

8. **Ouchani, Samir**, "A security policy hardening framework for socio-cyber-physical systems," *Journal of Systems Architecture*, vol. 119, p. 102259, 2021.

4.1 Fractionation-based Verification

To overcome the verification and modeling limitations, we propose to reduce the cost of the verification process of a system, modeled as SysML activity diagrams, by fractioning the initial diagram to ease the use of other abstraction, reduction, and refinement operations [31]. The overall framework developing the proposed solution is depicted in Fig. 4.1.

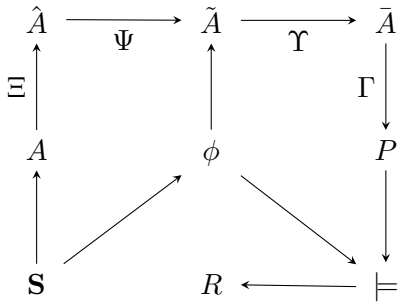


Figure 4.1: Fractionation-based Verification.

Starting from the specification of a given system **S**, the framework considers initially the SysML activity diagram A that models properly **S** and its requirement specification ϕ that is expressed as Probabilistic Computation Tree Logic (PCTL) to be guaranteed on A. Then, the function Ξ fractionates A to be $\hat{A}$ ($\hat{A}$ is the hierarchical representation of A) that is abstracted later using the

function Ψ with respect to ϕ, and the result is a new diagram $\tilde{A}$. To reduce more $\tilde{A}$, the reduction function Υ is applied to return a new compacted diagram $\bar{A}$. Further, to verify the satisfiability of ϕ, the transformation function Γ translates $\bar{A}$ into P, a PRISM source code. Finally, we demonstrate the practical application of the proposed framework using a case study that would be otherwise difficult to verify. Besides, we observe a significant reduction in the state space by an important rate, which makes probabilistic model checking helpful. We detail the framework flow depicted in Fig. 4.1 by presenting first the fractionation algorithm (Section 4.1). Then, the abstraction mechanism including the reduction rules (Section 4.1).

Fractionation

For an optimal hierarchical representation, first we look for the largest possible sub diagram called *ample*, a sub diagram with single input and single output, that can be extracted from the initial one. Then, we proceed the same process till satisfying the stopping criteria [1] (a sequence of opaque nodes). We fractionate A by extracting a set of sub-diagrams $A_1, \cdots, A_n$ satisfying the stopping criteria such that $A = A'[\mathtt{a}_1 \uparrow A_1, \cdots, \mathtt{a}_n \uparrow A_n]$ where ' = ' means both sides have equivalent behaviors.

To align within the standard syntax of SysML activity diagrams, an *ample* is bounded by a single input node and a single output node. Definition 13 stipulates formally an *ample* where $pred(a)$ and $succ(a)$ return respectively the predecessor and successor nodes of a given node a.

Definition 13. Let $A = \langle N,\ E,\ G,\ Grd,\ Prob\rangle$ be a SysML activity diagram, $A' = \langle N',\ E',\ G',\ Grd',\ Prob'\rangle$ is the largest ample of A such that $A = A''[\mathtt{a} \uparrow A']$ where $A'' = \langle N'',\ E'',\ G'',\ Grd'',\ Prob''\rangle$, iff:

- $N' \subseteq N$, $N'' = (N \setminus N') \cup \{\mathtt{a}\uparrow\}$, $\mathtt{a}_i = \mathtt{a}''_i \neq \mathtt{a}'_i$, and $\mathtt{a}_f = \mathtt{a}''_f \neq \mathtt{a}'_f$,
- $E' \subseteq E$ and $E'' = (E \setminus E') \cup \{pred_A(succ_{A'}(\mathtt{a}'_i)) \rightarrowtail \mathtt{a}, \mathtt{a} \rightarrowtail succ_A(pred_{A'}(\mathtt{a}'_f))\}$,
- $G' \subseteq G$, $G'' = G \setminus G'$, $Grd(E) = Grd'(E') \cup Grd''(E'')$ and $Grd'(E') \cap Grd''(E'') = \emptyset$, and
- $Prob''(N'') = (Prob(N) \setminus Prob'(N')) \cup \{pred_A(succ_{A'}(\mathtt{a}'_i)) \rightarrow_1 \mathtt{a}, \mathtt{a} \rightarrow_1 succ_A(pred_{A'}($

The algorithm "FRACTAL" based on rules of Definition 13 finds the largest possible amples of a given SysML activity diagram A by developing the function Ξ. It is called recursively to find a new ample inside the largest ample found by making it as an SysML activity diagram. The diagram is visited using a depth-first search procedure and the algorithm's output is a fractal diagram. First, the initial node is pushed into the stack of nodes denoted by *nodes* (line 6). The algorithm recursively pops a node from the stack *nodes* into the current node denoted by *cNode* (line 8). For a current node *cNode* with a single input edge (line 11) where its successor nodes have the same output node (line 13) the current node is considered and its marked successors are pushed into the stack of nodes (line 14). After

[1] Initially, we consider a path of opaque nodes as the smallest sub diagram.

all nodes are visited, the new diagram $newD$ to be called in the current node $cNode$ is constructed from $nNode$ by adding an initial and a final node (lines 21-24). Recursively, the algorithm is called for all obtained diagrams satisfying single input single output (lines 25-27).

By relying on the underlying semantics of activity diagrams proposed through operational rules and resulting a probabilistic automaton [33], we consider M_A as the probabilistic automaton of A; and we denote by R_Ξ the relation related to the function Ξ that is defined in Definition 14.

Definition 14 (Fractionation relation)**.** Let $A, A' \in A$ where $A' = \Xi(A)$; then, $M_A R_\Xi M_{A'}$ denotes the relation between the probabilistic automata M_A, $M_{A'}$ of A and A'.

The soundness of Ξ speculated in Lemma 1 shows that the relation R_Ξ is a bi-simulation relation.

Lemma 1 (Fractionation Soundness)**.** The fractalization algorithm Ξ is sound, i.e. $M_A R_\Xi M_{A'}$ is a bi-simulation relation.

Proof. For a given A where $A' = \Xi(A)$, we prove the soundness of Ξ by showing the correspondence between any state and its related transitions in M_A with its similar associated one in $M_{A'}$.

We implement Ξ by adding two Dirac transitions (Conditions 2 and 4 in Definition 13) having silent actions for each called diagram. Further R_Ξ is an equivalence relation since Ξ is reflexive, symmetric et transitive. And for any action in M_A we have the same in $M_{A'}$ and we have for every two states s and s' in M_A and M'_A the same probability since Ξ does not affect the probabilistic decisions. Then, M_A and M'_A are probabilistically bi-simulated. □

Abstraction

The abstraction step of the current framework implements, first Ψ then Υ, by extending the one developed in [33] and collapsing states that have similar behaviors as well by taking advantages from the properties of the operator $\uparrow$.

To implement Ψ, we consider a PCTL (Probabilistic Computation Tree Logic) expression ϕ to be verified on A where Σ_ϕ is the set of the atomic propositions of ϕ. By assuming $\Sigma_\phi \subseteq N$, we propose Definition 15 that reduces the size of A. The first rule excludes the nodes of the diagram that are unrelated to the activity whereas the second excludes the entire called diagram.

Definition 15. For a given SysML activity diagram $A \uparrow_{\mathtt{a}} A'$ and a PCTL expression ϕ such that $\Sigma_\phi \subseteq N$, we have

- $\forall \mathtt{a}_x \notin \Sigma_\phi \wedge \mathtt{a}_x \in N \cup N' : \Psi(\mathtt{a}_x \rightarrowtail N) = N.$
- $\Sigma_\phi \cap N_{A'} = \emptyset : \Psi(A \uparrow_{\mathtt{a}} A') = A.$

Further, Definition 16 develops the set of collapsing rules implemented by function Υ.

Definition 16. For a SysML activity diagram A, we define a set of reduction rules that are applicable on the artifacts $\|$, $|$, $\blacklozenge$, and $\Diamond$ as follows.

- $\Upsilon(\|(\mathtt{a}_1, \|(\mathtt{a}_2, \mathtt{a}_3))) = \|(\mathtt{a}_1, \mathtt{a}_2, \mathtt{a}_3)$,
- $\Upsilon(|(\mathtt{a}_1, |(\mathtt{a}_2, \mathtt{a}_3))) = |(\mathtt{a}_1, \mathtt{a}_2, \mathtt{a}_3)$,
- $\Upsilon(\blacklozenge(\mathtt{a}_1, \blacklozenge(\mathtt{a}_2, \mathtt{a}_3))) = \blacklozenge(\mathtt{a}_1, \mathtt{a}_2, \mathtt{a}_3)$,
- $\Upsilon(\Diamond_p(\mathtt{a}_1, \Diamond_{p'}(\mathtt{a}_2, \mathtt{a}_3))) = \Diamond_{p.p', p.(1-p'), (1-p).(1-p')}(\mathtt{a}_1, \mathtt{a}_2, \mathtt{a}_3)$,
- $\Upsilon(\Diamond_g(\mathtt{a}_1, \Diamond_{g'}(\mathtt{a}_2, \mathtt{a}_3))) = \Diamond_{g \wedge g', \neg g \wedge g', \neg g \wedge \neg g'}(\mathtt{a}_1, \mathtt{a}_2, \mathtt{a}_3)$.

The algorithm "AbsRed" illustrated in Algorithm 4 abstracts a given SysML activity diagram A by taking into account all atomic propositions of a specification ϕ '$Var(\phi)$'. The diagram is visited using a depth-first search procedure, then, the algorithm's output is a less complex diagram. First, the initial node is pushed into the stack of nodes denoted by *nodes* (line 5). The algorithm recursively pops a node from the stack *nodes* into the current node denoted by *cNode* (line 7) and adds each visited node into the list *vNode* of visited nodes (line 9). Then, it applies the abstraction rules in order to minimize the diagram A (lines 10-12). In each iteration, the destination nodes of the *cNode*'s outgoing edges are explored. When two successive nodes are matched (line 14), then they will be collapsed (line 15) to be pushed into *nodes* (line 21). The condition in line 17 excludes a diagram not overlapped with the atomic propositions of ϕ. The algorithm terminates when all nodes are visited.

The algorithm implements the composed function $\Upsilon \circ \Psi$ by calling first Ψ then Υ. Hence, Definition 17 defines the relation $R_{\Upsilon \circ \Psi}$ between the probabilistic automata of A denoted by M_A and $M_{A'}$ of A' obtained by $A' = \Upsilon(\Psi(A))$.

Definition 17 (Abstraction-Reduction relation)**.** Let $A, A' \in A$ where $A' = \Upsilon(\Psi(A)$; $M_A R_{\Upsilon \circ \Psi} M_{A'}$ is the relation between the probabilistic automata M_A, M'_A of A and A', respectively.

The soundness of the composed functions speculates the type of relation between M_A and $M_{A'}$ where $A' = \Psi \circ \Upsilon(A)$.

Lemma 2 (Abstraction-Reduction Soundness)**.** $\Psi \circ \Upsilon$ is sound, i.e. $M_A R_{\Upsilon \circ \Psi} M_{A'}$ is a probabilistic weak simulation relation.

Proof. For $A' = \Upsilon(\Psi(A))$, we prove the soundness of $\Upsilon \circ \Psi$ by showing the correspondence of a set of transitions in M_A with a weak transition in $M_{A'}$ for a given state in M_A.

We have $\Upsilon \circ \Psi$ implemented in Algorithm 4 hides actions by applying Ψ which replaces a set of transitions in M_A with a weak transition in $M_{A'}$. Further, Υ merges nodes that reduces states in $M_{A'}$ by preserving the probability distribution for the weak transitions. Then, for the transitions of any state in M_A there exist a corresponding weak transition for a state in $M_{A'}$ to represent the same behavior. Consequently, $M_A R_{\Upsilon \circ \Psi} M_{A'}$ is a probabilistic weak simulation relation. □

Algorithm 4 The Abstraction-Reduction Algorithm.

Input: SysML activity diagram A.

Output: New Abstracted and Reduced SysML activity diagram A'.

```
1:  nodes as Stack                                          ▷ All nodes..
2:  cNode as Node                                           ▷ Current node..
3:  nNode, vNode as list_of_ Node                           ▷ New and visited nodes..
4:  procedure AbsRed(M, φ)
5:      nodes.push(in);
6:      while not nodes.empty() do
7:          cNode := nodes.pop();                           ▷ Current node to analyze..
8:          if cNode not in vNode then
9:              vNode.add(cNode);
10:             if cNode ∉ InstOf(fin,in)∪Var(φ)) then
11:                 nNode := next(cNode);                   ▷ Abstracting nodes.
12:                 nodes.delete(cNode);
13:             end if
14:             if (Match(nNode,succ(nNode))) then
15:                 M.Collapse(nNode,succ(nNode));          ▷ Reducing.
                                                            ▷ nodes..
16:             end if
17:             if (Var(φ) ⊄ A') then
18:                 A.delete(A');                           ▷ Excluding a diagram..
19:             end if
20:         end if
21:         nodes.push(nNode);                              ▷ Constructing new diagram..
22:     end while
23: end procedure
```

4.2 Composition

Our compositional verification framework takes a set of SysML activity diagrams composed by the call behavior interface and a Probabilistic Computation Tree Logic (PCTL) [32] property as input. First, we develop an abstraction approach that restricts the verification of a PCTL property only on the influenced diagrams instead of the whole composition. Then, we propose a compositional verification approach by interface processes that distributes a PCTL property into local ones which helps to verify them separately for each diagram. For verification, we encode the diagrams into the PRISM input language [34]. Finally, we deduce the result of the main property from the results of the local properties that are verified separately for each called diagram.

Let A be a SysML activity diagram with n call behaviors denoted by $A = A_0 \uparrow_{a_0} A_1 \cdots A_{i-1} \uparrow_{a_{i-1}} A_i \cdots A_{n-1} \uparrow_{a_{n-1}} A_n$. In order to reduce the diagram A, we apply NuAC axioms and introduce the reduction rule defined in Definition 18 to remove diagrams A_i that are not influenced by the property

ϕ to be verified. The obtained diagram after applying the reduction rule is denoted by $\widehat{A}$.

Definition 18. Let A be a diagram that contains n call behaviors, AP_ϕ is the atomic propositions of the PCTL property ϕ, and AP_{A_i} is the atomic propositions of the behavioral diagram A_i. Reducing A to the diagram $\widehat{A}$ with respect to ϕ is obtained by applying the following rule.

$$\frac{\forall 0 \leq i \leq n, AP_\phi \cap AP_{A_i} = \emptyset}{A_i = \epsilon}$$

Below, Proposition 1 shows the satisfiability probability after reduction.

Proposition 1. For a reduced diagram $\widehat{A}$ of A with respect to ϕ, we have: $[\widehat{A} \models \phi] \Rightarrow [A \models \phi]$.

Proof. The proof of this proposition follows an induction reasoning on the PCTL structure. First, we take the case of $\psi = \phi_1 \mathrm{U} \phi_2$.
By definition, for $0 \leq i \leq n$ where $AP_\psi \cap AP_{A_i} = \emptyset$, then: $A_i = \epsilon$. The result is $\widehat{A} = A_0 \uparrow_{a_0} A_1 \cdots A_{k-1} \uparrow_{a_{k-1}} A_k$ and $k \leq n$.
From the PCTL semantics, we have $[(A_0 \uparrow_{a_0} A_1 \cdots A_{k-1} \uparrow_{a_{k-1}} A_k) \models \psi] \Leftrightarrow \exists m,\ \forall j < m : \pi(j) \models \phi_1 \wedge \pi(m) \models \phi_2$ where $\pi(j)$ and $\pi(m)$ are the states i and j respectively in a path π of A. And, by calling A_i in a_i using BH-1, the only changes in the path π are the propositions of A_i till executing BH-2, then: $\exists m' \geq m,\ j' \geq j,\ \forall j' < m' : \pi(j') \models \phi_1 \wedge \pi(m') \models \phi_2 \Leftrightarrow\ A_0 \uparrow_{a_1} \ldots \uparrow_{a_k} A_k \ldots \uparrow_{a_i} A_i \models \psi$.
By calling a new A_{i+1} in a_{i+1} up to n, we will have: $\exists m'' \geq m',\ j'' \geq j',\ \forall j'' < m'' : \pi(j'') \models \phi_1 \wedge \pi(m'') \models \phi_2 \Leftrightarrow A_0 \uparrow_{a_1} \ldots \uparrow_{a_n} A_n \models \psi \Leftrightarrow A \models \phi_1 \mathrm{U} \phi_2$.
For $\phi_1 \mathrm{U}^{\leq k} \phi_2$ and $\mathrm{X}\phi$ cases, we deduce the following.

- $\forall 0 \leq i \leq n, AP_\phi \cap AP_{A_i} = \emptyset : [A_i = \epsilon \wedge (A_0 \uparrow_{a_0} A_1 \cdots A_{n-1} \uparrow_{a_{n-1}} A_n) \models \phi_1 \mathrm{U}^{\leq k} \phi_2] \Rightarrow [\exists k' \geq k : A \models \phi_1 \mathrm{U}^{\leq k'} \phi_2]$.
- $\forall 0 \leq i \leq n, AP_\phi \cap AP_{A_i} = \emptyset : [A_i = \epsilon \wedge (A_0 \uparrow_{a_0} A_1 \cdots A_{n-1} \uparrow_{a_{n-1}} A_n) \models \mathrm{X}\phi] \Rightarrow [A \models \mathrm{X}\phi]$.

□

For a parallel verification, we decompose the PCTL property ϕ into local ones $\phi_{i:0 \leq i \leq n}$ over A_i with respect to the call behavior actions $a_{i:0 \leq i \leq n}$ (interfaces), we introduce the decomposition operator "$\natural$" proposed in Definition 19. The operator "$\natural$" is based on substituting the propositions of A_i to the propositions related to its interface a_{i-1} which allows the compositional verification. We denote by $\phi[y/z]$ substituting the atomic proposition "z" in the PCTL property ϕ by the atomic proposition "y".

Definition 19 (PCTL Property Decomposition)**.** Let ϕ be a PCTL property to be verified on $A_1 \uparrow_a A_2$. The decomposition of ϕ into ϕ_1 and ϕ_2 is denoted by $\phi \equiv \phi_1 \natural_a \phi_2$ where AP_{A_i} are the atomic propositions of A_i, then:

1. $\phi_1 = \phi([l_a / AP_{A_2}])$, where l_a is the atomic proposition related to the action a in A_1.

2. $\phi_2 = \phi([\top / AP_{A_1}])$.

The first rule is based on the fact that the only transition to reach a state in A_2 from A_1 is the transition of the action l_a (BH-1). The second rule ignores the existence of A_1 while it kept unchanged till the execution of BH-2. To handle multiplicity for the operator "$\natural$", we have Property 3.

Property 3. The decomposition operator $\natural$ is associative for $A_1 \uparrow_{a_1} A_2 \uparrow_{a_2} A_3$, i.e. :

$$\phi_1 \natural_{a_1} (\phi_2 \natural_{a_2} \phi_3) \equiv (\phi_1 \natural_{a_1} \phi_2) \natural_{a_2} \phi_3 \equiv \phi_1 \natural_{a_1} \phi_2 \natural_{a_2} \phi_3.$$

For the verification of ϕ on $A_1 \uparrow_{a_1} A_2$, Theorem 4.2 deduces the satisfiability of ϕ from the satisfiability of local properties ϕ_1 and ϕ_2 obtained by the operator $\natural$. [Compositional Verification] The decomposition of the PCTL property ϕ by the decomposition operator $\natural$ for $A_1 \uparrow_{a_1} A_2$ is sound, i.e. :

$$\frac{A_1 \models \phi_1 \qquad A_2 \models \phi_2 \qquad \phi = \phi_1 \natural_{a_1} \phi_2}{A_1 \uparrow_{a_1} A_2 \models \phi}$$

Proof. The proof of Theorem 4.2 follows a structural induction on the PCTL structure by using Definition 19. As an example, we take the until operator "U". Let $\phi = ap_1 \text{ U } ap_2$ where $ap_1 \in AP_{A_1}$ and $ap_2 \in AP_{A_2}$. By applying Definition 19, we have: $\phi_1 = ap_1 \text{ U } a_1$ and $\phi_2 = \top \text{ U } ap_2$. Let $A_1 \models \phi_1 \Leftrightarrow \exists m_1,\ \forall j_1 < m_1 : \pi_1(j_1) \models ap_1 \wedge \pi_1(m_1) \models ap_1 \wedge a_1$ where π is a path in the NuAC PA of A. For $A_2 \models \phi_2 \Leftrightarrow \exists m_2,\ \forall j_2 < m_2 : \pi_2(j_2) \models \top \wedge \pi_2(m_2) \models ap_2$. To construct $A_1 \uparrow_{a_1} A_2$, BH-1 is the only transition to connect π_1 and π_2 which form: $\pi = \pi_1.\pi_2'$ such that $\pi_2'(i) = \pi_2(i) \cup \pi_1(m_1)$. Then: $\exists j \leq m,\ m = m_1 + m_2 :\ \pi(j) \models ap_1 \wedge \pi(m) \models ap_2 \Leftrightarrow A_1 \uparrow_{a_1} A_2 \models \phi$. □

Finally, Proposition 2 generalizes Theorem 4.2 to support the satisfiability of ϕ on an activity diagram A with n call behaviors.

Proposition 2 (CV-Generalization)**.** Let ϕ be a PCTL property to be verified on A, such that: $A = A_0 \uparrow_{a_0} \cdots \uparrow_{a_{n-1}} A_n$ and $\phi = \phi_0 \natural_{a_0} \cdots \natural_{a_{n-1}} \phi_n$, then:

$$\frac{\begin{array}{c} A_0 \models \phi_0 \cdots A_n \models \phi_n \\ \phi = \phi_0 \natural_{a_0} \cdots \natural_{a_{n-1}} \phi_n \end{array}}{A_0 \uparrow_{a_0} \cdots \uparrow_{a_{n-1}} A_n \models \phi}$$

Proof. We prove Proposition 2 by induction on n.

- The base step where "$n = 1$" is proved by Theorem 4.2.
- For the inductive step, first, we assume:

$$\frac{\begin{array}{c} A_0 \models \phi_0 \cdots A_n \models \phi_n \\ \phi = \phi_0 \natural_{a_0} \cdots \natural_{a_{n-1}} \phi_n \end{array}}{A_0 \uparrow_{a_0} \cdots \uparrow_{a_{n-1}} A_n \models \phi}$$

Let $A' = A_0 \uparrow_{a_0} \cdots \uparrow_{a_{n-1}} A_n$ and $\phi' = \phi_0 \natural_{a_0} \cdots \natural_{a_{n-1}} \phi_n$. While $\natural$ and $\uparrow$ are associative operators, then: $A = A' \uparrow_{a_n} A_{n+1}$ and $\phi = \phi' \natural_{a_n} \phi_{n+1}$. By assuming $A_n \models \phi_n$ and applying Theorem 4.2, then:

$$\frac{\begin{array}{cc} A' \models \phi' & A_{n+1} \models \phi_{n+1} \\ A = A' \uparrow_{a_n} A_{n+1} & \phi = \phi' \natural_{a_n} \phi_{n+1} \end{array}}{A \models \phi}$$

□

4.3 Hardening

Reasoning about a system's security in combination with policies has been already explored and many models have been proposed such as role-based access control (RBAC), discretionary access control (DAC), location-based access control (LBAC), attribute-based access control (ABAC). Unfortunately, most of solutions do not take into consideration the policy impact on the required functionalities, especially in the case of complex and heterogeneous systems such SCPS. A high-level formalization of the analysis of an SCPS with a policy can be formulated as follows. Let S be a model of an SCPS, π be a policy supposed to constrain what can happen in the system by producing a more secure one $S_{|\pi}$ with respect to π, ϕ be a desirable functional requirment or a security property, and I be an adversary model. The question whether a security policy is effective in realizing a security goal can be symbolically represented by: $S_{|\pi} \models_I \phi$. Intuitively, assuming that all symbols used are instantiated appropriately, $S_{|\pi}$ represents the (execution of) S where π is enforced, while $\models_I \phi$ is the relation "satisfies ϕ in the presence of I". In absence of a policy, the question collapses into $S \models_I \phi$, the classical problem of determining whether S satisfies ϕ –or, informally stated, whether the SCPS is secure– in the presence of I. The aim is to automate this hardening and satisfiability issues in a precise formal framework for the security analysis of SCPS [15].

Definition 20 (Requirements/Policies Affectedness)**.** Let ϕ be a requirement and π be a policy and S a model of execution of a SCPS. Let $\texttt{traces}(S, \phi)$ be the set of traces in S that satisfy ϕ, and $\texttt{traces}(S, \neg\pi)$ the set of traces where π is *not* satisfied. We say that ϕ is affected by π in S, and we write $\phi \leftharpoondown \pi$ when $\texttt{traces}(S, \phi) \cap \texttt{traces}(S, \neg\pi) = \emptyset$.

Definition 21 illustrates an SCPS model constrained with a policy.

Definition 21 (Constrained CPS)**.** Let $S = \langle S, S_0, \Rightarrow \rangle$ to be an CPS, π a security statement represents a policy for S, and ϕ a security statement represents a requirement for S, then, S constrained with a policy π, written (S, π) is a new SCPS, $S' = \langle S', S_0, \Rightarrow' \rangle$ such that:

1. If $S \not\models \pi$ then $(S, \pi) \models \pi$;

2. For all π such that $\phi \not\leftarrow \pi$ then if $S \models \phi$ then $(S, \pi) \models \phi$;

Definition 22 (Constraining SCPS)**.** For a given $S = \langle S, S_0, \Rightarrow \rangle$ and a policy π, constraining S with π produces S' by executing three actions defined as follows.

1. *CleanS*() $\triangleq$ If $\exists s \in S$ where $s \models \phi$:
   - $S' = S \backslash \{s\}$, and
   - If $\exists (s', s) \in \Rightarrow$: $\Rightarrow' = (\Rightarrow \backslash \{(s', s)\}) \cup \{(s', s')\}$, and
   - If $\exists (s, s') \in \Rightarrow$: $\Rightarrow' = (\Rightarrow \backslash \{(s', s)\})$.
2. *CleanT*() $\triangleq$ If $\exists \rho = [s_i, \ldots, s_{j-1}, s_j]$ where $\rho \models \pi$:
   - $\Rightarrow' = (\Rightarrow \backslash \{(s_{j-1}, s_j)\}) \cup \{(s_{j-1}, s_{j-1})\}$.
3. *CleanR*() $\triangleq$ If $\exists s' \in S', \nexists (s'', s') \in \Rightarrow'$: $S' = S' \backslash \{s'\}$.

To produce the semantics of an SCPS S constrained with a policy ϕ, we parse, with a depth-first search, the SCPS transition system. Algorithm 5 does this search and constructs a policy constrained semantics of an SCPS S [7].

The algorithm first checks if a policy ϕ is of type ψ_{SP}, ψ_{PL}, or ψ_{TL}. In the case of a state statement, the procedure γ looks for states satisfying the policy ϕ to exclude them from the original CPS by calling the function *CleanS*(). For the case of path statements, the procedure finds the path that satisfies ϕ and calls the function *CleanT*(). Finally, the function *CleanR*() cleans the unreachable state. The function *CleanS*() replaces the incoming edges of a state with loops, and ignores its outgoing edges. Finally, it excludes the state from the set of states in S. The function *CleanT*() replaces the last transition of the path formula with a loop. The function *CleanR*() finds the predecessors of states, and it excludes them as well as its successors from the state space, and the set of transitions, respectively. *CleanR*() terminates when all states are reachable.

Now we enforce the model of the presented example with the policy $\Box(\neg(loc_A(I) \neq l_0))$ that claims an intruder should never access the infrastructure. Figure 4.3 shows only the retained steps on the CPS model presented after enforcing this policy. Further, we enforce again the model with a policy that claims if the object o_2 has been possessed by a_1, then, it will never be possessed by a_2. It is expressed as $\Box(\neg(\{o_2\} \subseteq cont_A(a_2) \rightarrow \Diamond(\{o_2\} \subseteq cont_A(a_1)))$. As a result, Figure 4.3 shows the added transition loop in green and the deleted one in red from the initial CPS formal model [35, 7].

Procedure γ (Algorithm 5) exhaustively searches all states and paths that satisfy a security policy. Functions *CleanS*() and *CleanT*() change the model in order to enforce it to satisfy a security policy. Function *CleanR*() cleans the model from unreachable states and avoids deadlocks. Algorithm 5 is sound means constrained models always satisfy the conditions given in Definition 21).

Algorithm 5 Policy Constrained Algorithm.

1: **procedure** $\gamma(S,\phi)$
2: **Input:**
1. $S = \langle S, S_0, \Rightarrow \rangle$; ▷ A tuple modelling an SCPS..
2. ϕ; ▷ A security statement..
3: **Output:**
1. $S = \langle S', S_0', \Rightarrow' \rangle$; ▷ A policy-based constrained CPS..
4: **if** $\phi.type() \in \{SP, PL\}$ **then** ▷ The case of state formula..
5: **for** each $s \in S$ **do**
6: **if** $s \models \phi$ **then**
7: $CleanS(S,s)$; ▷ Clean the states that satisfy ϕ..
8: **end if**
9: **end for**
10: **else if** $\phi.type() \in \{TL\}$ **then** ▷ The case of path formula..
11: **for** each $\rho \in 2^{\Rightarrow}$ **do**
12: **if** $\rho[s_i, \ldots, s_{j-1}, s_j] \models \phi$ **then**
13: $CleanT(S, s_{j-1} \Rightarrow s_j)$; ▷ Clean the last transition of the path ρ..
14: **end if**
15: **end for**
16: **end if**
17: $CleanR(S)$; ▷ Clean the unreachable states..
18: **end procedure**

Proposition 3 (Soundness)**.** If $S' = \gamma(S,\phi)$ is generated by Algorithm 5 for S and ϕ, iff, both conditions of Definition 21 hold.

4.4 Probabilistic Verification

For the analysis, we rely on the probabilistic symbolic model checker PRISM that verifies probabilistic specifications over probabilistic models. A specification can be expressed either in the probabilistic computation tree logic (PCTL) and a model can be described using PRISM language.

A model can be a discrete-time Markov chain, continuous-time Markov chains, and Markov decision processes (MDPs). Alternatively, a model can be probabilistic timed automata. PRISM also

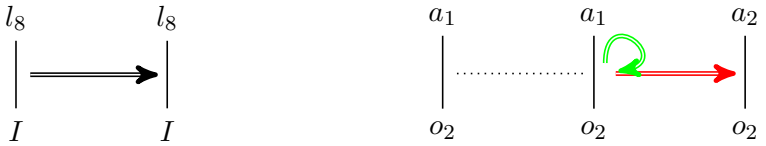


Figure 4.2: The effect of ϕ_1 (left) and ϕ_1 (right).

supports probabilistic automata. PRISM verification is efficient since it stores models as binary decision diagrams and multi-terminal binary decision diagrams (BDD). To overcome the state explosion problem, PRISM has built-in symmetry reduction and implements some iterative numerical analysis like Jacobi and Gauss-Siedel.

In general, a given PRISM program is a composition of a set of *modules*. A module is evaluated over a fixed number of local variables, of type Boolean or integer. Whereas the state of a given module is formulated as the evaluation of its local variables, the global state of a PRISM program is the evaluation of all variables, local and global, for all modules. Further, the composition and the communication between PRISM modules adopt the operators developed by the CSP process algebra [13].

Basically a PRISM module defines the kernel behaviour of a PRISM program. At this end, the behaviour of a module is a collection of commands that can be probabilistic or Dirac. Textually, a probabilistic command is expressed by $[\alpha]\ g \rightarrow p_1 : u_1 + ... + p_m : u_m$, such as p_i are probabilities ($p_i \in]0,1[$ and $\sum_{i=0}^{m} p_i = 1$), α is a label expressing the name of the action α, g is the guard represented as a propositional logic formula over all variables, local and global, and u_i describes the *update* (new value) for an ensemble of variables. A given update expressed by $(v'_j = val_j) \& \cdots \& (v'_k = val_k)$, assigns only values val_i to local variables v_i. So, for a given action α, if the guard g is valid, then the update u_i is enabled with a probability p_i. In general, the guard is an expression consisting of the evaluation of all variables that are connected explicitly with the propositional logic operators. The Dirac case where $p = 1$ is a special case command expressed simply by: $[a]\ g \rightarrow u$.

Syntactically, a module named M is delimited by two keywords: the module head "`module` M", and the module termination "`endmodule`". Further, we can model costs with a reward module R delimited by keywords "`rewards` R" and "`endrewards`". A reward module is composed from a *state reward* or a *transition reward*. A state reward associates a cost (reward) of value r to any state satisfying g and it is expressed by $g : r$. A transition reward is specified by $[a]\ g : r$ to express that the transitions labelled a, from states satisfying g, are acquiring the reward of value r.

Finally for the analysis, a PRISM program P proper will be generated to For that, we introduced the function T_P that assigns for each

For the semantic rule of any entity, its premises represent the guard of the entity PRISM command, whereas the update describes the consequence of the rule. For example, o_{o_2} is an atomic proposition showing the the object o possess o_2, l_a and l_o present the locations, and p_{o_3} precises the physicality attribute of o_3. The variables and propositions are evaluated first to describe the initial state of nodes by relying on the tuple obtained by the *Actuator* proper to each entity. T_P implements this transformation for each entity depends its category, and here we consider the transformation of rules presented in [26].

$$T_P(\alpha) = \begin{cases} [Syn_{o_2}]o_{o_2} \wedge o1_{o_3} \wedge \neg p_{o_2} \wedge \neg p_{o_3} \rightarrow (o'_2 = o_2); \\ [Syn_{o_2}]o_{o_2} \wedge o1_{o_3} \wedge \neg p_{o_2} \wedge \neg p_{o_3} \rightarrow (o'_3 = o_2); \\ \\ [Tak_{o_1}]l_a = l_o \wedge o_{o_2} \wedge \neg lock_o \wedge p_{o_2} \rightarrow (a'_{o_2} = \top); \\ [Tak_{o_1}]l_a = l_o \wedge o_{o_2} \wedge \neg lock_o \wedge p_{o_2} \rightarrow (o'_{o_2} = \bot); \\ \\ [loc_{o_1}]o_{o_1} \wedge o_{o_2} \wedge \neg k_{o_1} \wedge p_{o_1} = p_{o_2} \rightarrow (k'_{o_1} = \top); \\ [loc_{o_1}]o_{o_1} \wedge o_{o_2} \wedge \neg k_{o_1} \wedge p_{o_1} = p_{o_2} \rightarrow (o'_{o_1} = \top); \end{cases}$$

4.5 Statistical model checking

Building and verifying systems in BIP require three inputs: (i) describing the system's architecture in BIP language, (ii) providing a low-level code in C++ to model the probabilistic functions, (iii) and expressing the system's requirements in PBLTL. Then, BIP compiler is responsible for reading the user input (i.e. BIP source code) through a parser and produces the resulting system's code in C++. Finally, the composed source code is built by linking the external C++ files withing the produced ones. Hence, an executable artefact is generated for simulation.

The generated code could be feed to the stochastic engine where almost all probabilistic models are covered such as MDP. The stochastic engine encapsulates an executable model simulator and is used to produce (random) execution traces on demand. The monitor is used to evaluate properties on traces, and, then to produce local verdicts $\{true, false\}$. Moreover, the statistical model checkking (SMC) [2] engine implements the main statistical model checking techniques, namely, hypothesis testing [9] and probability estimation [12]. Finally, the parametric exploration module coordinates the evaluation of a parametric property. Also, SMC is brought with an integrated development environment including a graphical user-interface permitting to edit, compile, simulate models, and plotting graphs for parametric properties.

As mentioned earlier, queries/requirements to be verified using SMC-BIP shall be expressed in probabilistic bounded linear temporal logic (PBLTL). The syntax of the PBLTL temporal logic is expressed using the following Backus normal form (BNF) grammar.

$\varphi ::= true \mid ap \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \mathtt{P}_{\bowtie \mathtt{p}}[\psi]$
$\psi ::= N\varphi \mid \varphi_1 \cup^t \varphi_2 \mid F^t\varphi \mid G^t\varphi$

Here, "ap" is an atomic proposition, P is a probabilistic operator where $P_{\bowtie p}[\psi]$ means that the probability of a path formulae ψ being true always satisfies the bound $\bowtie p$, $\bowtie \in \{<, \leq, >, \geq\}$, where $p \in [0,1]$. "$\wedge$" and "$\neg$" represent the conjunction and the negation operators, respectively. Four path formulas are considered: the next operator $N\varphi$, bounded until $\varphi_1 \cup^t \varphi_2$ and bounded eventually

[2] http://www-verimag.imag.fr/BIP-SMC-A-Statistical-Model-Checking.html?lang=en

$F^t\varphi = true \cup^t \varphi$, and the bounded always $G^t\varphi = \neg(true \cup^t (\neg\varphi))$ where t is an integer value that specifies the length of the targeted system execution trace. φ_1 andφ_2 are state formula.

PBLTL semantics is defined with respect to an execution trace $\pi = s_0 s_1 \ldots$. Roughly speaking, an execution trace π satisfies $N\varphi$, which we denote $\pi \models N\varphi$, if s_1 satisfies φ. $\pi \models \varphi_1 \cup^t \varphi_2$ holds iff there exists a state s_i with $i \leq t$ that satisfies φ_2 and all the states in the prefix from s_0 to s_{i-1} satisfies φ_1. Using this query language, it is possible to formulate two more queries for a bounded LTL formula φ on a given stochastic BIP model.

- Qualitative queries : $\mathtt{P}_{\geq\theta}[\varphi]$, where $\theta \in [0,1]$.
- Quantitative queries : $\mathtt{P}_{=?}[\varphi]$, where φ is a bounded LTL formula.

Below are two illustrative examples with their natural language translation.

- $\mathtt{P}_{\geq 0.68}[\mathtt{F\ reboot}]$ "*The probability of the system eventually collecting information from sensors and reboot is at least 0.68*".
- $\mathtt{P}_{=?}[\mathtt{F\ shutdown}]$ "*What is the probability that the system eventually shutdown after collecting information from sensors ?*".

Based on the resulting BIP modelling, we rely on SMC-BIP to perform statistical analysis. SMC-BIP produces the needed run-timed traces to verify bounded LTL properties. The implemented simulators produce traces in different modes, i.e. symbols-wise, piece-wise and trace-wise. We use the first mode for online monitoring and to be able to interrupt simulations as soon as a verdict is obtained. The second simulator is primordial for rare events analysis whereas the third mode is dedicated to offline monitoring.

One of the feature keys of SMC-BIP is that it can find the probability of a specified LTL property holding on to the generated traces. For example, we can set a property to determine the probability that globally `train1` will not exceed the next stop while the velocity keeps respecting the recommended civil speed on the segments `DUBL_CAST_E` and `CAST_E_BAYF_S`:

$$\begin{aligned}\varphi_1 :\mathtt{P}_{=?}[\mathtt{G}^{\leq \mathtt{T}}\ (\mathtt{T1.position} <= (\mathtt{NEXT\_STOP} - \mathtt{stopDistance})\\ \&\&\ (\mathtt{T1.v} <= 36 || (\mathtt{T1.v} <= 80\ \&\&\ \mathtt{T1.v} >= 36))], \mathtt{T} = 1000\end{aligned} \tag{4.1}$$

The `NEXT_STOP` refers to the closed gate at `SANL (25428.1m)`, the stop distance is the required distance between the train and the segment gate. We use SMC-BIP with the confidence parameters α= `0.005` and δ = `0.05` for all our experiments (at this step and after). These confidence parameters require the evaluation of 199 system executions to come up with a global verdict, using the probability estimation technique. Property 4.1 is equivalent to Property 4.2 such that at some state of the

execution trace, the train's position is prior than the NEXT_STOP with a velocity less than the segment threshold, while at the next states of the trace the velocity is less than the second segment threshold.

$$\begin{aligned}\varphi_2 :& \mathrm{P}_{=?}[\ (\mathtt{T1.position} <= (\mathtt{NEXT\_STOP} - \mathtt{stopDistance}) \&\&\ (\mathtt{T1.v} <= 36) \\ & \mathrm{U}^{\leq \mathtt{T}}\ (\mathtt{T1.position} <= (\mathtt{NEXT\_STOP} - \mathtt{stopDistance})) \&\& (\mathtt{T1.v} <= 80 \\ & \&\&\ \mathtt{T1.v} >= 36)], \mathtt{T} = 1000\end{aligned} \tag{4.2}$$

The computed satisfiability probability is 1 for Property 4.1, meaning that on the generated traces the stopstopDistance is respected and the recommended civil speed is not exceeded.

SMC-BIP supports Parametric Exploration (PX) which performs a statistical model checking on a parametric property $\varphi(x)$, where x is an integer parameter ranging over a finite instantiation domain Π. Also, the simulation algorithm returns a set of SMC verdicts corresponding to the verification of the parametric property instances $\varphi(v_x)$ with respect to $v_x \in \Pi$. So, we can set a probability that inspects the train station arrival at SANL (25428.1m). This can be encoded with the following property:

$$\begin{aligned}\varphi_3(\mathtt{x}) :& \mathrm{P}_{=?}[\mathrm{F}^{\leq \mathtt{T}}\ \mathtt{T1.position} == (\mathtt{NEXT\_STOP} - \mathtt{stopDistance}) \\ & \&\&\ \mathtt{T1.CLOCK} <= \mathtt{x})], \mathtt{T} = 1000, \mathtt{x} = 0 : 400 : 10\end{aligned} \tag{4.3}$$

Meanwhile, it is necessary to augment the BIP model with a new instruction for the transition S3 → S4 by defining a new float variable CLOCK:

```
on computeAcceleation from S3 to S4 do { CLOCK = CLOCK + delta; }
```

The delta value is initialized to 0.5 as mentioned in Listing 4.1. So, the verification results of Property 4.3 using SMC-BIP is portrayed in Figure 4.3. It confirms that the position remains stable after 200 times units, which means that the train never exceeds the gate position at SANL (25428.1m). The result also confirms the requirement: "The train should not enter a closed gate".

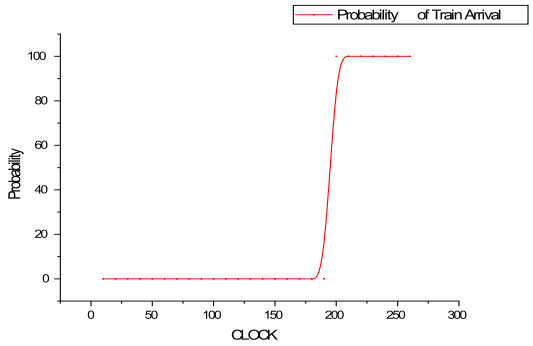


Figure 4.3: Probability of Train Destination Arrival for property 4.3.

Another main requirement stating that the trains on the same rail shall not collide which means

that the trains will keep a safe distance until the next destination. In that case, `train1` is initialized to a position `20m` with an initial velocity equals to 13.5 whereas `train2` is located at the starting point. This property is encoded as follows.

$$\varphi_4 : \mathtt{P}_{=?}[\mathtt{G}^{\leq \mathtt{T}}\ (\mathtt{T1.position} <= \mathtt{T2.position})\ \&\&\ \mathtt{T2.position} <= (\mathtt{NEXT\_STOP} - \mathtt{stopDistance})] \quad \mathtt{T} = 1000, \mathtt{x} = 0:400:10 \tag{4.4}$$

Property 4.4 expresses that while trains are progressing, the position of `train2` is greater than the position of `train1` in spite that `train2` is leading the rail progress. The computed satisfiability probability of Property 4.4 is `1.0`, meaning that the requirement " The train should not get so close to another train" is always satisfied.

Listing 4.1: Train Variables Initialization

```
1 initial to START do {
2       delta = 0.5;
3       nose = 10;
4       id=VID;
5 }
```

Listing 4.2: Train Components Instantiating

```
1 compound type Compound ()
2       component Train train1 (1)
3       component Train train2 (2)
4       component AATC aatc (1)
5       connector transfertTrainParameters(train1.
            transmitTainParameters, train2.
            transmitTainParameters, aatc.
            receiveTrainPositionAndAccAndVelocity)
6       connector transfertTrainVCMandACM(aatc.
            transmitAcmAndVcm,train1.receivingVMandACM,train2.
            receivingVMandACM)
7 end
```

4.6 Probabilistic and Network Simulation: smart city application

This section covers the analysis part of a smart city model [8, 36] includes both **P**hysical **M**odels (**PM**) and **D**igital **M**odels (**DM**). The analysis step checks and validates how well **SCM** models are

functionally correct through verification and simulation techniques. This step considers the developed **SCM** models as a network of Timed Automata (TA). Hence, the Uppaal model checker is used to simulate and check if the requirements are satisfied. Consequently, the Cooja network simulator previews if Wireless sensor networks (WSNs) achieve a low consummation of energy with high coverage of the area of interest.

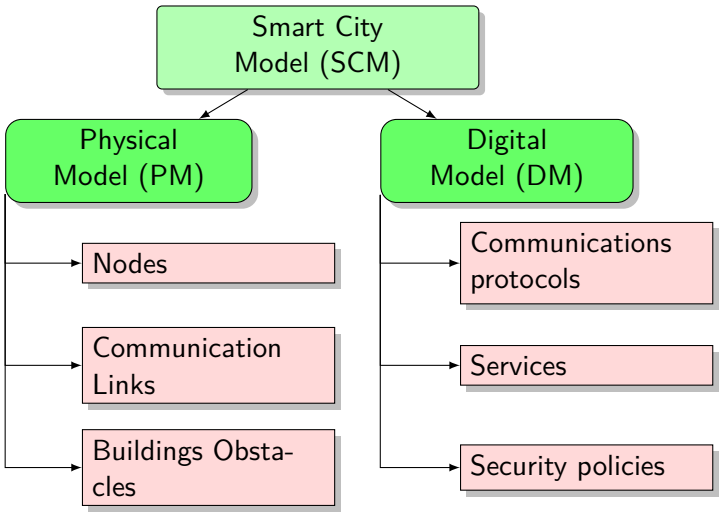


Figure 4.4: Smart City Model.

Smart City Model. We consider an **SCM** as an association that brings together both the digital and physical models (Figure 4.4). **SCM** architecture is divided into three levels. The third level is dedicated to processing and storage services by including different resources such as the servers and calculators with software to receive, process, and share data. Physically, there is a long distance between the first and the third level components, e.g. when the request is forwarded from the *third* level devices to the cloud computing server. The second level (**Communication**) is a collection of internet stations and providers to link the other levels. The first level ($Sensing and Action$) is the indoor sub-architecture secured by hardware and software tools. It contains unconstrained devices, that are responsible to monitor and request data (like computers and smartphones) through different protocols.

The Physical Model (**PM**) is a set of hard components that visually construct the concrete building/city. The Digital Model (**DM**) is a collection of digital components and rules to guarantee the functional correctness of **ICTs**. The proposed **DM** covers the adopted protocols, services, and security protocols. *Nodes* are a set of sensing, application, processing, routing, and storing appliances such as sensors, actuators, servers, routers, and data center. A given $Node$ by the tuple $\langle attr, action, State, Behavior \rangle$, where: $attr$ is a set of *static* and *dynamic* attributes evaluated by the value val. The "*static*" attributes are fixed while a node is running, *e.g.* the size of an object, memory

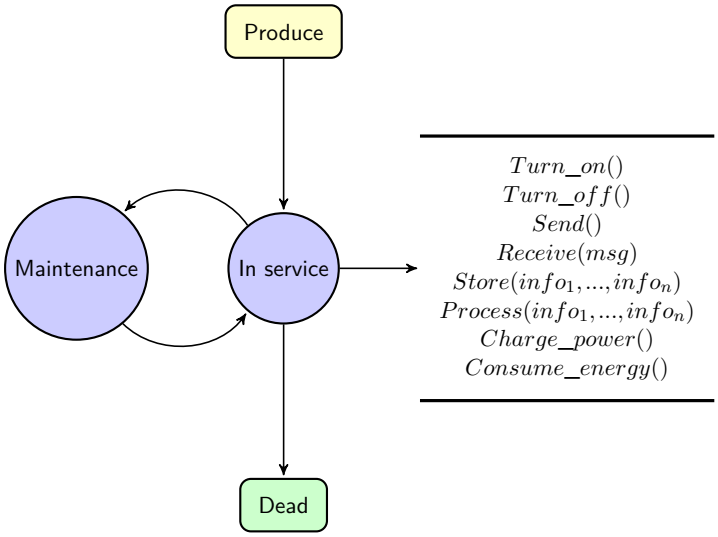


Figure 4.5: Cycle life of $Node$.

capacity, etc. The *dynamic* ones change when a node executes its proper *actions*, e.g, the battery degree, availability (On/Off), etc. The evaluation of *attr* by *val* can be real or boolean. A given $Node$ can execute a predefined actions during its life cycle (see Figure 4.5).

Figure 4.6 shows the architecture of our concrete **SCM** that we want to analyze. It is a client/server architecture based on RPL(Routing Protocol for Low-Power and Lossy Networks), Constrained Application Protocol (CoAP), and MQTT (Message Queueing Telemetry Transport) protocols. The third level represented by the processing unit is equipped by the cloud computing server that records less-used information (e.g, buildings status report per week) and the fog computing service which stores frequently the most used information (e.g, the measured data). Further, it has the ISP that supports the wire and wireless communication. In this architecture, we consider unconstrained and constrained devices; the unconstrained devices are the communication, filtering, routing and protecting appliances (computers, firewall, routers and the IDSs respectively). The constrained devices play the role of the fire detection system (fire sensor, broker, and an actuator that spray the water into the emergency case). The fire system nodes communicate through MQTT protocol.

Network simulation. In this experiment, all the BIM sensors use RPL protocols to transmit the temperature measured in the buildings. The Multi-path Ray-tracer Medium (MRM) model is an extension chosen to simulate the presence of obstacles. By following the proposed architecture guidelines that avoid constructing the global network which helps to reduce the resources use of Contiki OS computer container. We divide the global network into multi sub-networks related by sinks. Then, the RPL protocol constructs a graph of routes (DODAG, Destination-Oriented Directed Acyclic Graphs) using the Minimum Rank with Hysteresis Objective Function (MRHOF) algorithm.

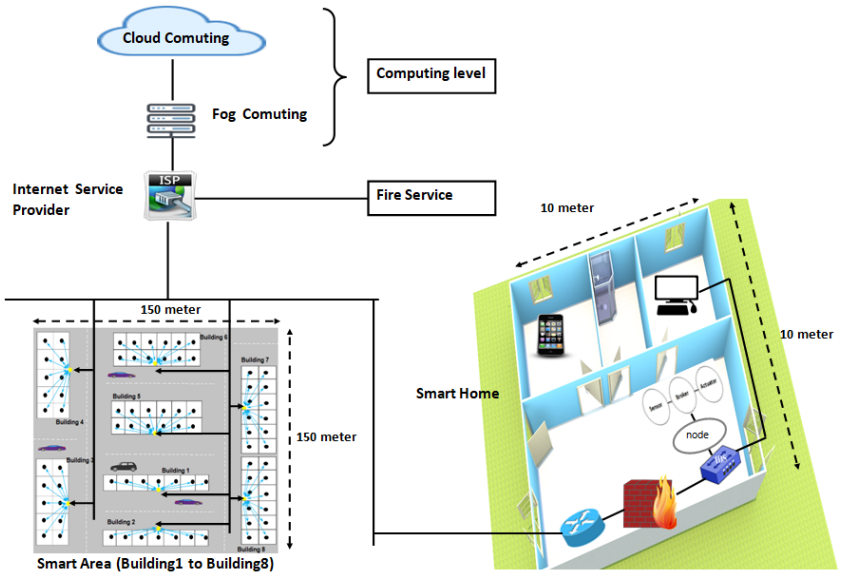


Figure 4.6: Area of interest: Smart City.

Figure 4.7 illustrates the probability of receiving the signal of one sensor in the area of interest (sensor 3, building 1).

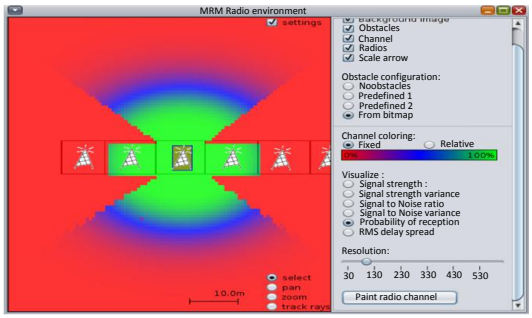
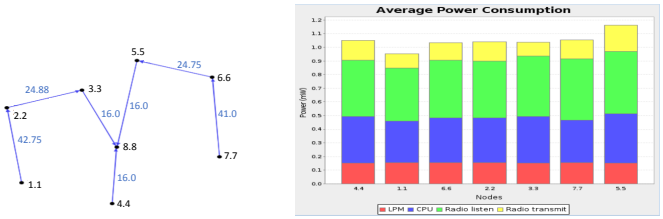


Figure 4.7: Probability of receiving signals.

From the simulation results, we found that any WSN recognizes its neighbours to construct the DODAG. During 5 minutes of simulation, the nodes in each building constructs its DODAG, where the sink is the meeting point of all orientations. We observe that all WSNs are presented and connected to transmit the collected data to the sink. For example, $node_1$ represented in the DODAG of Figure 4.8(a), located in the first home of the Building 1, is far from the *sink* ($node\,8$) and its wireless communication passes through many obstacles. Thus, it has the greatest value (42) compared to others. After the connectivity insurance, we analyze the energy consumption of nodes in each building. Figures 4.8(b) illustrates the energy consumed in all buildings nodes concerning the number of executed operations: sensing (red color) by using he laser precipitation monitor (LPM), processing (blue color) by using a central processing unit (CPU), receiving using a radio listener (green color), and sending

by using a radio transmitter (yellow color).



(a) DODAG. (b) Consummation of energy.

Figure 4.8: Results in building 1.

Probabilistic simulation. First, we aim to monitor the fire alarm system and to analyze the fire case resulting in smart buildings and also to check the reaction of the IoT nodes in the network.

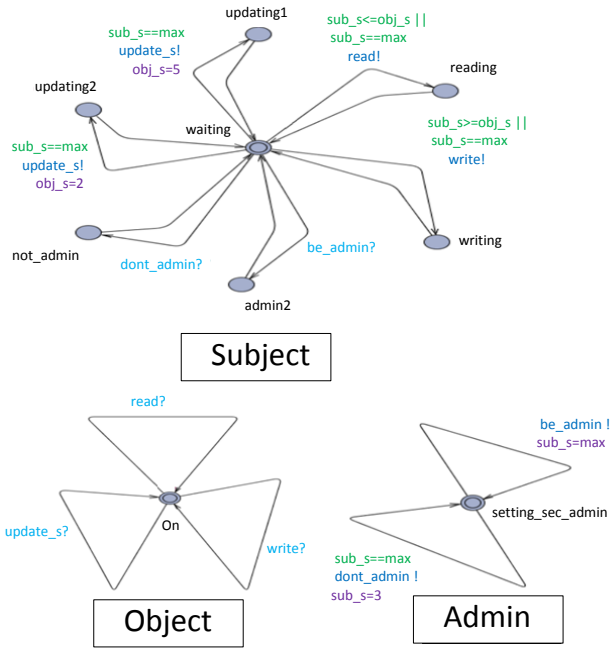


Figure 4.9: TA of ACM scenario.

The proposed fire alarm system contains three main components: the sensor which monitors and sends an alarm to the broker in the fire case, the broker sends the command (stop the fire) to the actuator that is subscribed on it, and the subscribed actuator in the broker receives the command from it. We relate the bropker by another node to inform the fire service (e.g., message describing the location of the building and the time of the incident). As a second step, we test our proposed access control model, where, we model three automata: *Admin*, *Subject* and *Node*. The role of *Admin* is to set the security level in *Subject* and *Object*, the *Subject* randomly can be *Admin* or $Non-Admin$.

The *Subject* applies actions to *Object* according to the security level of the *Subject* and *Object*. All security properties (**read, write and access**) are respected according to the alternative security level of the components presented in Figure 4.9.

4.7 Conclusion

After modeling **SCPS** and specifying its security and reliability under the presence of smart attacks, we have enhanced their analysis by introducing network and statistical simulation, and also by developing abstraction, and compositional verification techniques for probabilistic model checking. The experimental results have been shown the validity of the implemented solutions on benchmarks and critical systems, especial railways automotives and industrial use cases.

Chapter 5

Research Projects and Future Works

Content

In the previous chapters, we have presented our contributions about presenting a semantics to **SCPS** semantics (chapter 3). Then, we showed how to specify, reinforce and evaluate their security and reliability (chapter 4). Finally, we advanced the state-of-the-art by enhancing model checking approaches for **SCPS** (Chapter 5). In this last chapter, we present our research project for the coming years by applying, extending, and enhancing the developed approaches based on formal methods and artificial intelligence techniques.

5.1 Resiliency meets Security in SCPS (RSS-CPS)

Context. Socio-Cyber-Physical Systems (SCPS) are complex and heterogeneous systems that control large physical infrastructures [7]. They integrate smart and autonomous entities in addition to physical elements (e.g. actionners, sensors), informational elements (policies, norms, data), computational elements (e.g. procedure, communication protocols), and social actors (e.g. agents, operators). SCPS do not operate in isolation from the physical, digital, and social environments where they are displaced; on the contrary, they continuously receive stimuli from those environments. Despite that, they have to be robust and resilient to misuses, abuses, and intrusions. Further, due to their device proliferation and the overlap between the recommended standards, SCPS become emergent and

non-deterministic running in imprecise and dynamic environments, ruled by complex relationships, and constrained by organizational objectives. However, *S*CPS are expected to be resilient, immune to cyber-attacks and free of errors, which is practically difficult in real-life daily systems. Unfortunately, their strong dependency on connectivity enables the support of novel communication protocols and distance control functionalities; thus, it expands their attack surfaces, resulting in high risks to internal, external, and composed cyber-attacks.

Ensuring resiliency and assessing security in *S*CPS is a complex undertaking task since the challenge is guaranteeing the security requirements, fulfilling and adopting standards, and harmonizing the functionality between different smart parts that remotely interact with a range of different technologies.

Challenges, Hypothesis, and Objectives. Certainly, the discussed qualities are not easily measurable neither maintainable due to many factors: autonomous entities, smart attacks, dynamic environments, legal and economic impacts, etc. Recently, many initiatives based on formal methods and data-driven techniques have been developed but independently leveraged on large-scale systems. Both paradigms highlighted the significance of resiliency and cyber-security by showing how to evaluate, prevent, and mitigate breaches [37]. To bring both techniques together for *S*CPS, new challenges need to be addressed. The **first** is to include all the heterogeneous elements that compose *S*CPS. Different aspects need to be combined consistently into a harmonious design. Today, there are no solutions that offer a unified approach to model and analyze *S*CPS and consider *S*CPS' heterogeneity and smart elements together including the social dimension. The **second** is to include realistic threat and fault models. In *S*CPS, an adversary that tries to corrupt the elements on which the security of *S*CPS depends upon can resort to strategies of social engineering deceptive tricks and smart attacks. The **third** is to perform a security analysis that could be more informative than the worst-case analysis in security protocol. The traditional approach often shows that a system is insecure since very rarely complex systems are invulnerable. However, one would like to understand other parameters to qualify better the discovered attacks. Despite that, the **fourth** is to detect then mitigate all possible combinations of failures and vulnerabilities with respect to the requirements of the system under test and the adopted resiliency and security standards. However, providing a prevention mechanism that predicts attacks and failures and overcomes errors in real-time is the target of RSS-CPS.

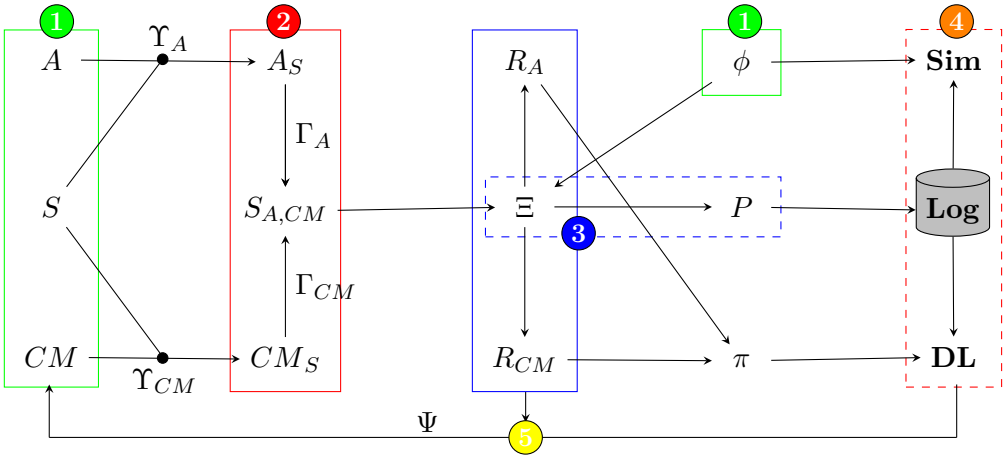
Existing approaches try to categorize such attacks and to detect vulnerabilities earlier but with many limitations. In fact, there is no clear initiative that models *S*CPS including smart entities with a social dimension. Security analysis and resilience assurance of *S*CPS is still in the first stage of applying formal methods with data mining techniques. The properties are specified according to the formalism supported by the tool in use, and the complexity of the *S*CPS model is not tackled properly. Providing security policies and modeling attacks with respect to smart entities of *S*CPS including their features is a real challenge that is not tackled yet. RSS-CPS project is about developing tools to support security analysis and resilience assurance for *S*CPS. RSS-CPS aim is to define resiliency and security requirements of *S*CPS, classifies the possible vulnerabilities and faults in *S*CPS entities, to

predict and analyze potential attacks, and to propose pertinent countermeasures that can by applied to mitigate the discovered attacks and errors. Further, RSS-CPSefficiently analyzes security for SCPS by proposing a sound analysis framework by combining formal methods and machine learning techniques. To take on the adversary mindset, RSS-CPS relies on hardening and deep-learning techniques in-order to predict faults before occurring and provide an emergency recovery plan through a smart recommendation system.

RSS-CPS investigates a practical and theoretical framework that enables security assessment and resiliency reinforcement for SCPS by achieving the following objectives. [❶] To capture the underlying concepts of SCPS with an adequate semantics, the first objective of RSS-CPS is to categorize and improve the existing modeling techniques and stereotype them for SCPS in order to produce a standardized SCPS modeling approach. Also, [❷] RSS-CPS aims developing a sound and automatic framework, by relying on formal methods together with machine learning techniques, to ensure the correctness and gauge how well a SCPS is secure and resilient.In addition, it improves the resilience of SCPS against dynamic environment conditions and cyber-attacks by developing a smart and dynamic recommendation system. [❸] Proposing optimal and scalable techniques for the previous objectives. [❹] Enabling a large library of security and resilience templates, including: security properties, policies, attacks, recovering plans and countermeasures. [❺] Implementing and releasing software supporting the developed methodology. [❻] Validating and testing the proposed concepts and tools on real-life industrial case studies.

Methodology. RSS-CPS analyzes and hardens the security aside from assuring the resiliency of SCPS. As input, it considers a SCPS as a composition of many entities with different aspects and behaviors that interleave in different environments and communicate through distinct ranges of protocols to realize complex tasks. RSS-CPS eases designing the main components of SCPS including their ways of communication through predefined editable templates. Further, it develops a library of attacks that can manipulate and harm the SCPS besides countermeasures that harden the system and counterfeit the attacks. Furthermore, RSS-CPS provides a library of security, safety, and resilience requirements, conditions and attributes. Thus, RSS-CPS can efficiently check and analyze the vulnerabilities and weaknesses of SCPS, along with tracing errors and generating counterexamples. In addition, RSS-CPS provides a recovery mechanism and a recommendation system that ensures the functional continuity of SCPS in case of system failures or security threats. When the modelled SCPS is certified to be safe, secure, and error-free; RSS-CPS generates the SCPS safe code that can be executed and simulated. Thence, in real-time, RSS-CPS monitors the SCPS and predicts the possible parasites, errors, faults, and attacks along with their corresponding countermeasures. The figure below depicts the overall five stages developing RSS-CPS initial approach as work packages: ① Design, ② Specialization, ③ Verification and certification, ④ Simulation and prediction, and ⑤ Correction and recovery.

At the design stage (①), S models a given SCPS, which is composed of *smart* entities of distinct aspects that interact through different ways, albeit A and CM are libraries of attacks and countermeasures proper to S, respectively [7]. The formulae ϕ expresses resilience, safety, and security properties



and constraints [38]. Then, in the second stage of composition (2), the function Υ_A sorts out the potential attacks A_S from A based on the possible attack surfaces in S, and the function Υ_{CM} selects from CM the countermeasures CM_S of S with respect to the selected attacks A_S [35]. Further, the function Γ_A composes S with A_S to produce a malicious system S_A, and the function Γ_{CM} hardens S_A with CM_S and produces $S_{A,CM}$ [17]. At the analysis stage (3), Ξ represents a smart and distributed verification module that checks at which degree S is satisfying ϕ by producing two results R_A and R_{CM} which show respectively the effect of A and CM on S [32, 31]. Both results can take the form of a quantified fault tree, graph, or/and independent counterexamples. When S is secure and certified safe ($S \vdash \phi$), Ξ generates P. The latter is executed in real-time (4) to produce traces (**Log**) that can be used by a simulation module (**Sim**) in order to monitor the SCPS behavior and requirements (ϕ). Continuously, it consolidates a recommendation system (5) based on deep learning module (**DL**) that learns the errors (π) and the execution traces (**Log**) in order to predict attacks and to provide the countermeasures that should be deployed before incidents [39, 17]. Based on the obtained results, from the third stages and the recommendations of the fourth stage, the function Ψ proposes the possible actions and procedures to be taken with respect to SCPS requirements(ϕ) and standards.

This project relies on our previous work of security modeling [40], specification [38], and analysis [17] for cyber physical systems [15]. In addition to our collaborators and partners, we are invited also to extend network and working with our colleagues in LIRMM, Montpelier.

5.2 Safety Assurance in Autonomous Transportation Systems (SAAS)

Context. Since the birth of Industry 2.0, road transport has evolved, and the automotive market has grown rapidly due to stiff competition. Consequently, pollution and safety remain significant issues. Subsequently, with the arrival of Industry 4.0 and then 5.0, the connectivity and autonomy

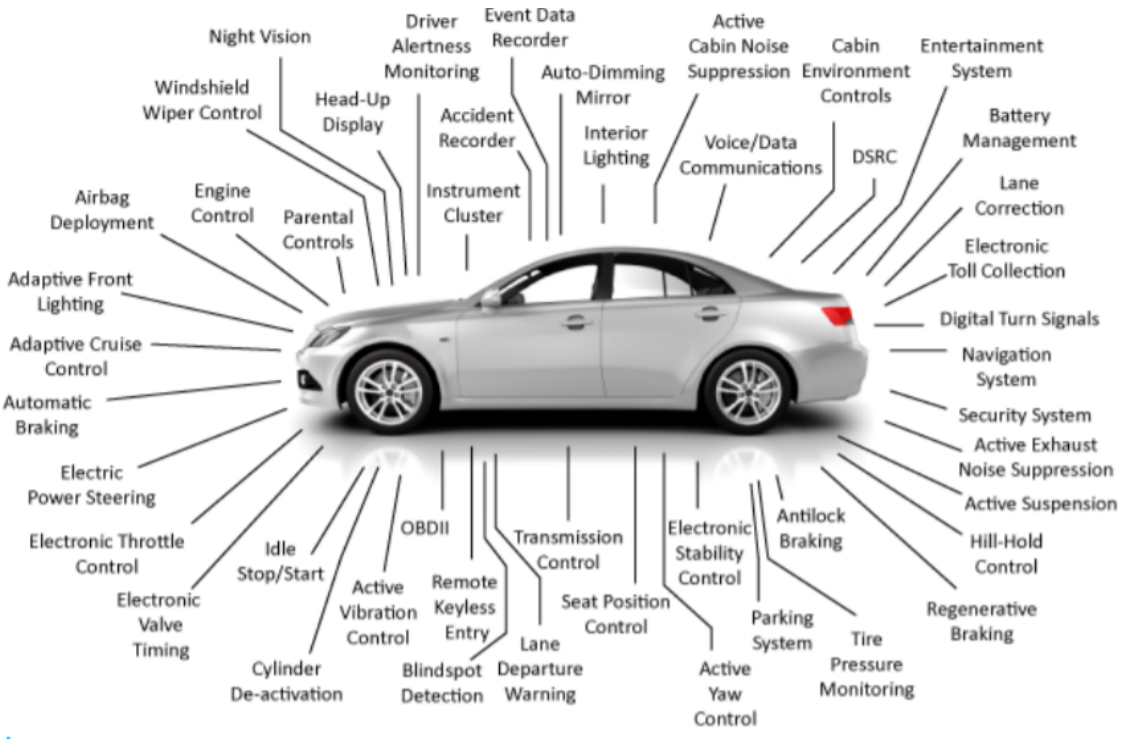


Figure 5.1: Automatic Functions of an Autonomous Vehicle.

of vehicles offered potential and transformative solutions in order to have safer journeys, optimal exploitation of the road network, and also a cleaner environment with reliable mobility.

Autonomous Vehicles (VA) have several driving assistance mechanisms that help the driver or even take control of the pilot when performing specific tasks such as "automatic parking". Such functionalities are made possible through the use of onboard computers, called Electronic Control Unit (ECU), which control the functionalities of a car. Thanks to a set of sensors and actuators distributed throughout the vehicle according to the type of tasks. For better control, the ECUs exchange data via communication buses forming an on-board oriented network. Likewise, the functions devolved to the software lead to an increase in the complexity of the functions and the risk of faults appearing during their design and subsequently their operation. Such faults can then be the cause of failures, the consequences of which can seriously affect the integrity of the vehicle and the safety of passengers. As a safety measure, the ISO 26262 standard provides methods for designing electrical and electronic systems that are safe to operate. However, the issues addressed by these methods mainly relate to safety and do not consider possible malicious interventions on embedded systems.

Like all types of computer systems, communication buses and ports constitute the first attack surfaces on the network of the onboard system. In the event of a successful attack, the intrusion will be able to read, modify and send messages on the communications buses to take control of the operation of the ECUs. Beyond that, *a security policy must be put in place with an overall view of*

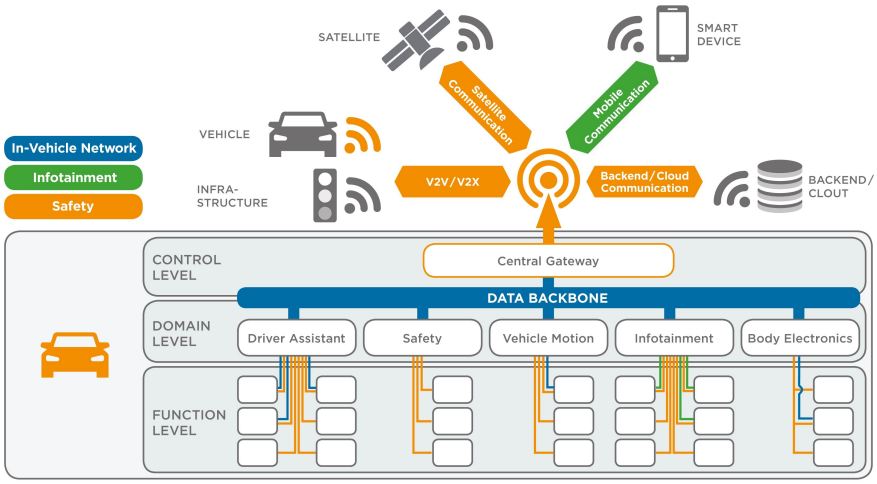


Figure 5.2: Autonomous Vehicle Network.

the system to be protected. Being satisfied with preventive security measures leaves the network of onboard systems vulnerable. It is, therefore, necessary to identify the targeted and compound actions of the attack. Intrusion detection also involves detecting security policy violations that occur on the observed system. A safety mechanism designed for an automobile must therefore remain efficient during its development, deployment, and operation. Likewise, a security mechanism must therefore be easily portable from one architecture to another so that the ECUs must operate autonomously and transparently for the user.

This project aims to set up intelligent mechanisms for the safety and immunity of automotive systems.

Challenges. ECUs are designed, like any on-board system, to meet particular needs, such as quality management, physical constraints, lifespan, limited resources, and real-time. In addition, the software embedded in each car ECU was the subject of ad-hoc development. In order to manage the complexity of the embedded systems developed, the automotive industry is turning to a more structured approach to software development. It has incorporated model-based design, such as Automotive Modeling Language (AML), COmponent Language (COLA), EAST -ADL, Timing Augmented Description Language (TADL), and ICT MAENAD.

In addition, thanks to connectivity, ADAS (Advanced Driver Assistance Systems) offers intelligent functions and allows more and more tasks related to driving to be automated. This connectivity has grown with the arrival of cooperation between vehicles, based on communications between nearby vehicles, known as V2V (Vehicle to Vehicle), and between a vehicle and fixed roadside infrastructure equipment, known as V2I. (Vehicle to Infrastructure). As shown in Figure 5.2, the notion of automotive networks encompasses a much larger whole than the network of onboard computers.

From a safety side, ECUs can be responsible for critical vehicle malfunctions, although they are

subject to safety and safety constraints referred to in ISO 26262. This defines criticality levels (ASIL, Automotive Security Integrity Level) for each element of an on-board system and offers different design approaches and mechanisms to add to ensure the system's proper functioning. Attacks against embedded systems are numerous and target the owner, the manufacturer, or even the government. Therefore, an analysis of the risks associated with security-immunity incidents in automobiles makes it possible to define a more effective defence strategy according to the needs expressed. To this end, [41] has proposed a classification of ECUs combining safety and security aspects via the introduction of SELs (Safety Effect Levels of security threats), which are used to assess the impact of a type of attack on the security of the on-board system. In addition, [42] proposed to guarantee the confidentiality of the location to mobile users through an architecture that develops: an epoch system, a labelled threat model based on the transition, and a query measuring the sensitivity of the location. Also, [43] analyzed safety and security by integrating a six-step method according to ISO 26262 and SAE J3061 standards. Thus, [44] showed the relation between a threat, an attack, a vulnerability, and its impact on an autonomous vehicle. Finally, [45] designed an architecture to secure sensitive areas from any suspicious activity of an autonomous vehicle by relying on navigation in a recursive path.

Unfortunately, there is not yet a standard describing an equivalent classification to ASIL that considers safety and immunity incidents. Vehicle safety is most important because lives depend on it, and system malfunction is not the only possible consequence of an attack. Attacks can target data integrity and access, such as data privacy. Therefore, the design of safety mechanisms for these automotive systems must consider autonomy and compatibility, including backward compatibility and interoperability. To do this, the safety mechanisms must be as autonomous as possible and only require the driver's attention when the situation demands it.

Methodology and Objectives. The objective of this project is the implementation of security-immunity means in automotive networks, and more specifically, concerns the detection, prediction, and forecasting of errors within these networks. Considering the lifespan of a car and the frequency of updates, it is unrealistic to deal with attacks simply with those that are known during the development of the car. Therefore, an effort must be made to achieve effective error detection even against attacks that will be discovered in the future in order to perform system updates. Thus, to guard against a wide variety of threats, a security policy should not be based entirely on a single type of protection means but use a mechanism acting at various levels (software, systems, network, and environment) from systems to protect.

Unfortunately, there is no standard describing the application of automotive safety policies nor corresponding certifications. The needs in terms of safety and immunity of automobiles are still relatively recent, and relatively little data is available on the subject. To meet these needs, we propose a hybrid approach between model and data, which ensures the development of a safe system and at the same time analyzes the deployed system and prevents errors and attacks based on the collected data. This approach aims to develop the following objectives:

1. A modelling language dedicated to autonomous vehicles based on existing languages and systems modelling standards (SysML, UML, OCL, etc.).
2. Classify the threats that can affect onboard systems in automobiles and the ECU network.
3. Security deployment standards.
4. A defence-in-depth and complementary defence mechanism to best protect a complex system such as a modern automobile against intrusion. This mechanism should adapt relatively easily to these various architectures.
5. A model-oriented risk analysis approach based on formal methods.
6. A data and model-oriented error detection approach based on formal methods and artificial intelligence techniques.
7. An artificial intelligence-based prevention and remediation approach has been proven in-depth on uncertain systems with large-scale data.

With our collaborators, we have initiating the application of our previous research activities towards the safety of autonomous vehicles, one of which [6, 40] initiative was developed targeting objectives 1 to 5. We target to hire a PhD student for this project with Concordia University.

5.3 Security and Safety meets Schedulability in Smart Healthcare Systems (SAFETY)

Context. Today, our world is facing at least three major challenges regarding the public health: the increase number of aging or elderly persons, the population growth, and the increasing prevalence of severe diseases. This leads to complicate the healthcare systems management with the emergence of COVID-19, a significant gap is noticed between the available hospital resources (i.e. medical centers, number of beds and qualified personnel) and those in need. The integration of new technologies in healthcare has become an essential task in order to allow hospitals to manage large numbers of patients while ensuring effective monitoring and rapid treatment when an emergency is detected. Hence, a suitable solution that handles these challenges are more than mandatory in the current situation. To increase the healthcare of indoor inhabitants and ensure their safety, we target to develop a more robust IT architecture powered with a smart recommendation system [39]. This solution should be useful at any time especially during crisis periods.

Recently, the emergence of sensing-based devices, especially the Wireless Sensor Network (WSN) and the Internet of Things (IoT), has led to a new revolution in healthcare [46, 47]. Mainly, this revolution is based on a set of biomedical sensors that continuously monitor vital signs (such as heart

rate, respiratory rate, oxygen level, temperature, blood pressure, etc.) of a patient staying at home (i.e. remote monitoring) and to periodically transmit the collected data to the hospital for a later analysis. Therefore, when a patient's critical condition has been detected, an emergency service intervention has been put in place to transfer the patient to the hospital and closely monitor his condition. This will allow, on the one hand, to reduce the number of patients in hospitals and conserve hospital resources and, also, to efficiently monitor the status of patients in real time and act accordingly.
Based on the fact of the above challenges, SaFeTy project looks to develop solutions that ensure the safety of patients and their efficient monitoring, as well as better manage the healthcare resources by relying on recent and prominent technologies, mainly Blockchains for security and data privacy, AI for decision making, Fog for data pre-processing, and IoT and WSN for sensing and network reliability.

Challenges and Objectives. The objective of this project, named SaFeTy, is to propose a robust and secure management system that ensures the safety of patients and helps the medical staff to efficiently master the pandemic crisis. This project aim is to address the following challenges.

I. Data management: Often, the biomedical sensors must operate autonomously for a long period of time, they have limited resources (memory, processing, and energy) and suffers from a low security protection. The first concern of SaFeTy project is as follows.

1. **Network architecture:** A healthcare system should be secure, robust, and scalable to react efficiently in real-time. Indeed, all information could be stored in a decentralized and secure fashion. In addition, all sensors should be permanently connected and accessible. Thus, new technologies are needed to be deployed especially blockchains, edges and fogs, and sensors [48]. Blockchains to maintain security, edges to ensure the robustness, and fogs to handle the network scalability. Designing such architecture is not an easy task, with respect to the literature and the existing ones, while it should satisfy the requirements recommended by different standards.

2. **Data storage and processing:** It is known that the exchanged data in healthcare systems is huge and sensitive. At this level of difficulty, the sensed measures are submitted in different forms and stored with different formats which lead to big data issues [9]. Unfortunately, the latter complicates data analysis and decision making. By relying on the network architecture with a decentralised fashion, our focus will be on how to design a robust distributed storage while using some dedicated solutions, such as Hadoop and its ecosystem. Consequently, machine learning techniques (mainly deep learning) will be highly investigated to help the medical staff to understand the received data, extract useful information, and make the right decisions according to the patient's status. As a decision support, the project looks to develop a prediction strategies based on machine learning that allows medical teams to optimally manage the pandemic situation and to detect in real time the active health diseases [49]. Further, it helps to monitor the virus propagation and prevent a possible contamination/infection to prioritize the medical care of patients. Based on this solution, the medical system will be able to predict with accuracy the future development of epidemic and enhance the patient care scheduling.

3. **Security and Privacy:** Healthcare data is highly sensitive and vital to patients and public services. Data must be transmitted and stored in a secure and protected way between different entities of the network architecture and its IT solution. Also, the solution should prevent and predict attacks by developing appropriate access control mechanisms and countermeasures [50]. At this end, the proposed blockchain technology allows to manage the access controls in transparency, secure the communication by adopting proper security communication protocols, and preserving the data privacy through a distributed solution.

4. **Reducing the energy consumption:** As previously mentioned, saving the available energies of sensors is essential for long-term patient monitoring. In addition, the collected data is inherently massive in terms of detection, processing, then transmission. To avoid periodic sensing, developing new AI-based algorithms becomes essential to reduce the amount of transmitted data and conserve the available energy in the sensors [51]. Also, basing on the provided data analysis techniques and categorizing the users profiles, energy management can be optimized more efficiently. Consequently, the collected data frequency depends on the identified criteria related to the patient health status and his historic.

SaFeTy project improves distant-medical consulting, assures security, privacy and integrity of stored and exchanged data that is collected from heterogeneous sensors. Based on more developed technologies like blockchains, edges, and fogs in addition to intelligent decision making techniques, our smart solution might detects in real time the active health deceases, predicts the virus propagation, and reacts with the best decisions.

II. Emergency service challenges: To intervene at time in emergency situations, one of the most recurrent problems is traffic. So, looking for the best route to achieve patients, it is essential to take into consideration all exceptional events, planned or unforeseen, that may affect normal traffic conditions [52]. For this reason, an alternative use of UAVs (unmanned aerial vehicle) to transport medical equipment is to be explored. The aim of this part is to develop a smart emergency service which couples the current solution based on the search for road routes with the routes of the UAVs. Indeed, we try to answer the following questions.

1. **Ideal locations for UAV centers:** Finding the suitable locations requires going back upstream of the problem to collect and organize all the necessary data (urban, road, hospitals, rescue workers) using the Geographic Information System (GIS). This will make it possible to extract the necessary geographic information and project it via a multi-criteria analysis grid to propose one or more suitable sites for UAV centers.

2. How can exceptional conditions (weather, disasters, occasional traffic jams, accidents) be taken into account when looking for a route? (Here it is necessary to define all the criteria to be taken into consideration and define the methods for collecting the information and updating it in real time). Choose according to the situation (decision tree) the adequate method for the intervention of the emergency service ((rescuers + drones) or ambulance).

3. How can travel time estimates be adjusted if there are changes in the physical characteristics of the road network?

Our basic idea is to design a system that communicates, in real time, with several actors to collect weather conditions, integrate information respectively related to exceptional events planned (such as demonstrations) and unforeseen (such as accidents on the road), integrate the physical characteristics of the road network, the vital state of the patient, the geolocation of rescue workers, the availability of the nearest hospitals, the degree of urgency to indeed calculate the best path relative to the situation.

III. Resources scheduling challenges: Unfortunately, due to its rapid spread and lack of immunization, until end of November, 2020, COVID-19 affected more than 60 million people and caused the death of one and half million others. However, hospitals did not have sufficient resources (rooms, beds, nurses and doctors) to assist infected patients where a number of countries lost control over this epidemic, especially European countries. This led to an exponential increase in the number of deaths in these countries and forced some governments to implement unethical healthcare policies, as a living age priority, to treat infected people [49, 47]. In this project, we aim to propose a dynamic allocation scheduling that helps to better exploit hospital resources, especially:

1. Manage the physical resources of a hospital: our methodology aims to propose tools and techniques that help in the programming of all care in a context of diminished capacities (imposed period of unoccupancy of wards/rooms between two patients, additional time for disinfecting equipment, limiting the number of patients in the waiting room, etc.) in a perspective of long-term modification of care capacities. Also, these tools and techniques will help manage the availability of beds in the hospital structure.

2. Managing a hospital's human resources: our methodology also aims to propose a scheduling or planning strategy that manages the timetables of the medical staff. This strategy takes into consideration staff capacities, standard conditions, and the case of attritions due to epidemic stress.

Keywords. Pandemic; Remote Monitoring; Patient Management; Real Time Data Analysis; Machine Learning; Emergency Case Detection; Security and Blockchain; Nurse Scheduling Strategies.

Based on our previous experience on anomalies detection [49, 47], decentralized systems [17, 20], and the secure communication between constrained objects [19], we are invited to collaborate with our colleagues from UHA.

5.4 Smart Federation of Mining Strategies in Decentralized ICPS (DFL)

Context. Modern industrial systems are known for their decentralized and distributed architecture since plants are geographically distant. Further, all components are rich in terms of heterogeneous sensors and strongly connected through different communication protocols. Including blockchains, different data based architectures are designed to deal with this kind of system and the huge amount of collected data. This technology ensures the security and data privacy of the different system components that depend on different mining strategies. Thus, deploying a smart decision system for processing and a applying a mining strategy needs to rely on machine learning algorithms, especially reinforcement learning.

Objectives. The first objective of this project is to study the existing decentralized architecture in industrial and production systems that rely primarily on Industry 4.0. Then, the second objective is to study the different machine learning techniques that can be used for which type of architecture. The objective of this project follows the next strategy:

1. Surveying the existing decentralized architectures in industrial systems.
2. Studying and comparing the existing machine learning algorithms.
3. Studying and comparing the mining algorithms and protocols.
4. Categorizing which machine learning algorithm is more appropriate to which specific architecture and application.
5. Mining optimization while federating the learning for a node.
6. Experiencing and deploying a selected group of techniques for a specific architecture and applied on a real production system.

This project has been started recently by initiating three master thesis in collaboration with ESI, Algeria.

5.5 Model-Based Equivalence: One to Many (OM)

Context. Modern systems are complex to design and to ensure their functionalities. Many formalisms are dedicated to the system design such as: Automata, Petri Nets, SysML, UML, etc. Further, different techniques are developed for analysis depends on the used formalism at the design phase; ex. model checking, theorem proving, static analysis, etc. Hence many tools are implemented carrying out a specific model.

Challenges. From a development perspective, is a selected tool can support other formalisms? This project looks to find the possible equivalence between models that will offer the use of a precise tool for a large modeling methodologies. For example a communication protocol can be seen as a UML sequence diagram, a composition of synchronized automata, or a set of SysML activity diagram. In this case, the thesis looks to find the equivalence between these formalisms in order to get the use of only one tool.

Objectives. This project targets to unify the existing modeling and verification tools by:

- Enumerating and comparing the existing, and developing when not found, modeling formalism dedicated to systems and software. Then, identifying their related analysis tools.
- Founding the existing relation between the resulting models.
- Showing and proving how this relation preserves the properties of each model.
- Developing a framework that implements this relation and gets advantage from the existing solutions.
- Showing the effectiveness of the developed framework on different types of models representing different application eras.

Keywords. Systems Modeling, Automata, Petri Nets, SysML, UML, Equivalence and Simulation Relation, Theorem Proving, Model Checking.

This project extends a part of my PRI (Projet de Recherche Individuel à Lineact, Individual Research Project at Lineact) to be driven with colleagues from Virginia University.

5.6 Safe and Smart Living (SSL)

Context. IoT is the interaction of physical objects -devices, vehicles, buildings and other items embedded with electronics, software, sensors, and network connectivity- that enables to collect and exchange massively data. This technology of intelligent device-to-device communication provides the much-needed leverage to IoT which make it grow extensively. It promises immense potential for improving the quality of life, health care, manufacturing, transportation, etc. The rise of IoT is not changing widely while using the same technology, connectivity, and trimmed mobile applications. Hence, IoT privacy is challenging due to the heterogeneity among devices, the massive exchanged data, and it relies on the same service providers and communication protocols.

Challenges. The design of IoT-based systems is complicated with the presence of different kinds of requirements in a mixed hardware-software environment. Not only must we assure that the system

will always behave safely and be protected against attackers, but we must also consider the end-users privacy by providing a mechanism that allows communicating a large amount of data in a proper way against different misuses of access privileges or attacks. Many modelling and analysis approaches like data mining, abstraction and refinement modelling, and formal methods help to detect flaws earlier and can analyze absolutely every possible situation, but none that really covers data privacy within the same modelling and data analysis technique especially the interaction between hazards and attacks while elaborating the hardware-software architecture of a system.

Objectives. The main objective of this project is to provide a smart interaction mechanism that preserves data privacy of objects while ensuring the system's requirements and respecting the system's partitioning. To achieve this goal, the thesis should focus on the following stages:

- Surveying the existing methodologies in (1) modelling and analyzing IoT systems, (2) IoT communication protocols, (3) analyzing massive data in IoT, and (4) describing and enforcing data privacy.
- Comparing the studied state of the art and enhancing the real challenges of privacy in IoT, and running a case study showing how to preserve massive data in an IoT application).
- Proposing a formalism that models precisely IoT and expresses very well data privacy. The selected formalism is based on the studied state of the art, especially the ones relying on formal methods and data analysis.
- Showing how to describe privacy and how to model IoT, deal with massive data, and how to express privacy in that system. These contributions envelop the first contribution chapter after the related work one.
- Proposing a framework that preserves data privacy in IoT that allows the description formalism developed previously. It shows and proves how data requirements can be violated before reinforcement and ensured after, and how they are preserved against attacks.
- Showing how to preserve privacy of data in IoT, and providing the tool and prove the correctness of the proposed preservation mechanism.
- Applied the contributions on case studies and providing a prototype.

Keywords: IoT, Privacy, Security Protocol, Data mining,Set Modulo Theory, Formal methods.

This work extends the thesis about security by design and its applications on smart cities.

5.7 General conclusion

In this dissertation, I have presented some of the research that I carried out at the University of Luxembourg (Luxembourg) and Lineact CESI (France). This work mainly concerned three aspects, formal semantics, security and reliability, and formal verification of **SCPS**. My research activities rely mainly on formal methods and their leveraging towards **SCPS** security, and also, with the orientation to reinforce such techniques by using artificial intelligence approaches.

The works presented in this manuscript are primarily supported by sound theoretical foundations, then, implemented and experimented on real use cases and benchmarks. This hat of both sides, theoretical and empirical, are clearly identified in our research projects. To have different point of views and more critics about our solutions, intentionally, we make the code source of our solutions available to the community. Thanks to this policy, we have had feedback and comments on our solutions that allowed us to improve it and sometimes fix bugs. We were also able by exchanging on our code, explain to users how well make use of it which allowed the adoption of our approaches. Our policy of making the code available is therefore helpful and practical, and continuing in this direction helps us to make our solutions more applied.

The different research projects in which I am involved open perspectives on other subjects and application like the fusion of enhancing formal methods by artificial intelligence. Also, the diversity of applications and partners in the various ongoing projects will enrich the fields of application of our approaches. Table 5.1 summarizes the different described projects, the identified calls and the partners. Further, Table 5.2 shows how my current research and the perspective projects develop, extend, and apply my expertise area in different topics and applications.

Project	Calls	Partner	Duration
RSS-CPS	ANR	CNRS LAAS, Concordia University	4 years
SAAS	FNR	SnT	3 years
SAFETY	PHC	Verimag, University of Blida	4 years
DFL	CIFRE	Evina	3 years
OM	Interreg	Verginia University, RMIT, LIRMM	3 years
SSL	PHC/Interreg	Lebanon University	2 years

Table 5.1: Projects summary and perspectives.

Thanks to the dynamic created by Lineact CESI laboratory and CESI Engineering School enable me to collaborate with my colleagues and their networks which gave us better visibility in the local and international institutional or industrial community. This new context has led to an increase in contacts which results on various collaborations and initiating new research projects.

Project	Semantics	Security	Reliability	Analysis
RSS-CPS	✓	✓	✓	✓
SAAS		✓	✓	✓
SAFETY		✓		✓
DFL		✓		✓
OM	✓			✓
SSL		✓		

Table 5.2: The impact of my current research activities on projects in perspectives.

Bibliography

[1] T. Van Hardeveld and D. Kiang, *Practical Application of Dependability Engineering: An Effective Approach to Managing Dependability in Technological and Evolving Systems.* ASME Press, 2012.

[2] C. Heitmeyer, M. Archer, E. Leonard, and J. McLean, "Applying formal methods to a certifiably secure software system," *IEEE Transactions on Software Engineering*, vol. 34, no. 1, pp. 82–98, 2008.

[3] L. Li, J. Sun, Y. Liu, M. Sun, and J.-S. Dong, "A formal specification and verification framework for timed security protocols," *IEEE Transactions on Software Engineering*, vol. 44, no. 8, pp. 725–746, 2018.

[4] Y. Chevalier, R. Küsters, M. Rusinowitch, M. Turuani, and L. Vigneron, "Extending the dolev-yao intruder for analyzing an unbounded number of sessions," in *Computer Science Logic* (M. Baaz and J. A. Makowsky, eds.), (Berlin, Heidelberg), pp. 128–141, Springer Berlin Heidelberg, 2003.

[5] D. Gollmann, C. Herley, V. Koenig, W. Pieters, and A. Sasse, "Socio-technical security metrics (dagstuhl seminar 14491)," *Dagstuhl Reports*, vol. 4, pp. 1–28, 01 2015.

[6] Abdelhakim Baouya, O. A. Mohamed, **Samir Ouchani**, and D. Bennouar, "Reliability-driven Automotive Software Deployment based on a Parametrizable Probabilistic Model Checking," *Expert Systems with Applications*, p. 114572, 2021.

[7] **Ouchani, Samir**, "A security policy hardening framework for socio-cyber-physical systems," *Journal of Systems Architecture*, vol. 119, p. 102259, 2021.

[8] Dahmane Walid, Miloud, **Ouchani, Samir**, and H. Bouarfa, "Towards a reliable smart city through formal verification and network analysis," *Computer Communications*, vol. 180, pp. 171–187, 2021.

[9] Baouya, Abdelhakim, S. Chehida, **Ouchani, Samir**, S. Bensalem, and M. Bozga, "Generation and verification of learned stochastic automata using k-nn and statistical model checking," *Applied Intelligence*, Nov 2021.

[10] Baouya, Abdelhakim, O. A. Mohamed, D. Bennouar, and **Ouchani, Samir**, "Safety analysis of train control system based on model-driven design methodology," *Computers in Industry*, vol. 105, pp. 1–16, 2019.

[11] **Samir, Ouchani**, *A Security Verification Framework for SysML Activity Diagrams*. PhD thesis, Concordia University, Faculty of Engineering and Computer Science, Montreal, Canada, September 2013.

[12] Abdelhakim, Baouya, O. Ait Mohamed, D. Bennouar, and **Samir, Ouchani**, "A formal approach for maintainability and availability assessment using probabilistic model checking," in *Modelling and Implementation of Complex Systems*, pp. 295–309, Cham: Springer International Publishing, 2016.

[13] Abdelhakim, Baouya, O. Ait Mohamed, D. Bennouar, and **Samir, Ouchani**, "Safety analysis of train control system based on model-driven design methodology," *Computers in Industry*, vol. 105, pp. 1–16, 2019.

[14] Baouya, Abdelhakim, D. Bennouar, O. A. Mohamed, and **Ouchani, Samir**, "A Formal Approach for Maintainability and Availability Assessment Using Probabilistic Model Checking," in *Modelling and Implementation of Complex Systems*, pp. 295–309, Springer, 2016.

[15] **Ouchani, Samir**, "Towards a security reinforcement mechanism for social cyber-physical systems," in *The Third International Conference on Smart Applications and Data Analysis for Smart Cyber-Physical Systems (Invited Paper)*, p. 14, LNCS, Springer, 2020.

[16] Abdelhakim, Baouya, "Bip model for bart system," 2021. https://github.com/hakimuga/TrainTransportSystem/blob/main/TrainTransportation.bip.

[17] K. Abd El-Aziz, **Samir, Ouchani**, T. Zahir, and D. Khalil, "Assessing the Severity of Smart Attacks in Industrial Cyber Physical Systems," *Transactions on Cyber Physical Systems*, 2020.

[18] **Ouchani, Samir** and G. Lenzini, "Generating attacks in sysml activity diagrams by detecting attack surfaces," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–13, 2015.

[19] Fahem, Zerrouki, **Samir, Ouchani**, and H. Bouarfa, "Towards a foundation of a mutual authentication protocol for a robust and resilient puf-based communication network," *Procedia Computer Science*, vol. 191, pp. 215–222, 2021. The 18th International Conference on Mobile Systems and Pervasive Computing (MobiSPC).

[20] Dahmane Walid, Miloud, **Ouchani, Samir**, and H. Bouarfa, "Guaranteeing information integrity through blockchains for smart cities," in *International Conference on Model and Data Engineering*, pp. 199–212, Springer, 2021.

[21] Zerrouki, Fahem, **Ouchani, Samir**, and H. Bouarfa, "A generation and recovery framework for silicon pufs based cryptographic key," in *International Conference on Model and Data Engineering*, pp. 121–137, Springer, 2021.

[22] F. Zerrouki, **S. Ouchani**, and H. Bouarfa, "Quantifying security and performance of physical unclonable functions," in *2020 7th International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, pp. 1–4, 2020.

[23] Zerrouki, Fahem, **Ouchani, Samir**, and H. Bouarfa, "A low-cost authentication protocol using arbiter-puf," in *International Conference on Model and Data Engineering*, pp. 101–116, Springer, 2021.

[24] B. Kim, S. Yoon, Y. Kang, and D. Choi, "Puf based iot device authentication scheme," in *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1460–1462, IEEE, 2019.

[25] R. Genser, "Critical infrastructure," in *Improving Stability in Developing Nations through Automation 2006* (P. Kopacek, ed.), IPV–IFAC Proceedings Volume, pp. 49–53, Oxford: Elsevier, 2006.

[26] Abdelhakim, Baouya, O. A. Mohamed, **Samir, Ouchani**, and D. Bennouar, "Reliability-driven automotive software deployment based on a parametrizable probabilistic model checking," *Expert Systems with Applications*, p. 114572, 2021.

[27] J. Carlson, J. Håkansson, and P. Pettersson, "Saveccm: An analysable component model for real-time systems," *Electronic Notes in Theoretical Computer Science*, vol. 160, pp. 127 – 140, 2006. Proceedings of the International Workshop on Formal Aspects of Component Software (FACS 2005)Proceedings of the International Workshop on Formal Aspects of Component Software (FACS 2005).

[28] I. Meedeniya, B. Buhnova, A. Aleti, and L. Grunske, "Reliability-driven deployment optimization for embedded systems," *Journal of Systems and Software*, vol. 84, no. 5, pp. 835 – 846, 2011.

[29] G. Heiner and T. Thurner, "Time-triggered architecture for safety-related distributed real-time systems in transportation systems," in *Digest of Papers: FTCS-28, The Twenty-Eigth Annual International Symposium on Fault-Tolerant Computing, Munich, Germany, June 23-25, 1998*, pp. 402–407, 1998.

[30] Abdelhakim, Baouya, "Bip model of aatc system with availabilities," 2021. https://github.com/hakimuga/TrainTransportSystem/blob/main/AATCwithAvailabilities.

[31] **Ouchani, Samir**, "Towards a fractionation-based verification: application on sysml activity diagrams," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pp. 2032–2039, 2019.

[32] **Ouchani, Samir**, "Towards a call behavior-based compositional verification framework for sysml activity diagrams," in *International Colloquium on Theoretical Aspects of Computing*, pp. 216–234, Springer, Cham, 2019.

[33] **Ouchani, Samir**, O. A. Mohamed, and M. Debbabi, "Efficient probabilistic abstraction for sysml activity diagrams," in *International Conference on Software Engineering and Formal Methods*, pp. 263–277, Springer Berlin Heidelberg, 2012.

[34] Baouya, Abdelhakim, D. Bennouar, O. A. Mohamed, and **Ouchani, Samir**, "A quantitative verification framework of sysml activity diagrams under time constraints," *Expert Systems with Applications*, vol. 42, no. 21, pp. 7493–7510, 2015.

[35] K. Lounis and **Ouchani, Samir**, "Modeling Attack-Defense Trees Countermeasures using Continuous Time Markov Chains," in *Automated and verifiable Software sYstem DEvelopment (ASYDE), Software Engineering and Formal Methods (SEFM)*, LNCS Springer, 2020.

[36] Dahmane, Walid Miloud, **Ouchani, Samir**, and H. Bouarfa, "A smart living framework: Towards analyzing security in smart rooms," in *International Conference on Model and Data Engineering*, pp. 206–215, LNCS Springer, 2019.

[37] Y. Ashibani and Q. H. Mahmoud, "Cyber physical systems security: Analysis, challenges and solutions," *Computers & Security*, vol. 68, pp. 81–97, 2017.

[38] **Ouchani, Samir**, O. A. Mohamed, and M. Debbabi, "A security risk assessment framework for sysml activity diagrams," in *2013 IEEE 7th International Conference on Software Security and Reliability (SERE),*, pp. 227–236, IEEE, 2013.

[39] A. Khaled, **Ouchani, Samir**, and C. Chohra, "Recommendations based on semantic analysis of social networks in learning environments," *Computers in Human Behavior*, vol. 101, p. 435 449, 2019.

[40] **Ouchani, Samir** and A. Khaled, "A meta language for cyber-physical systems and threats: Application on autonomous vehicle," in *16th International Conference on Computer Systems and Applications AICCSA*, p. 8, ACS/IEEE, 2019.

[41] J. Moody, N. Bailey, and J. Zhao, "Public perceptions of autonomous vehicle safety: An international comparison," *Safety Science*, vol. 121, pp. 634 – 650, 2020.

[42] J. Joy and M. Gerla, "Internet of vehicles and autonomous connected car - privacy and security issues," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–9, July 2017.

[43] J. Cui, G. Sabaliauskaite, L. S. Liew, F. Zhou, and B. Zhang, "Collaborative analysis framework of safety and security for autonomous vehicles," *IEEE Access*, vol. 7, pp. 148672–148683, 2019.

[44] S. Plosz and P. Varga, "Security and safety risk analysis of vision guided autonomous vehicles," in *2018 IEEE Industrial Cyber-Physical Systems (ICPS)*, pp. 193–198, May 2018.

[45] M. F. Ayub, F. Ghawash, M. A. Shabbir, M. Kamran, and F. A. Butt, "Next generation security and surveillance system using autonomous vehicles," in *2018 Ubiquitous Positioning, Indoor Navigation and Location-Based Services (UPINLBS)*, pp. 1–5, March 2018.

[46] **Samir Ouchani** and M. Krichen, "Ensuring the Correctness and Well Modeling of Intelligent Healthcare Management Systems," in *The Impact of Digital Technologies on Public Health in Developed and Developing Countries - 18th International Conference, (ICOST)*, pp. 364–372, LNCS, Springer, 2020.

[47] N. Dimeglio, S. Romano, A. Vesseron, V. Pelegrin, and **Ouchani, Samir**, "Covid-detect: A deep learning based approach to accelerate covid-19 detection," in *International Conference on Model and Data Engineering*, pp. 166–178, Springer, 2021.

[48] Walid Miloud Dahmane, M.-E.-A. Brahmia, J.-F. Dollinger, and **Samir Ouchani**, "A BIM-based framework for an Optimal WSN Deployment in Smart Building," in *The 29th International Conference on Network of the Future (NOF)*, IEEE, 2020.

[49] H. A. Le Thi, **Ouchani, Samir**, *et al.*, "Gene selection for cancer classification using dca," in *International Conference on Advanced Data Mining and Applications*, pp. 62–72, Springer Berlin Heidelberg, 2008.

[50] **Ouchani, Samir**, "Ensuring the functional correctness of IoT through formal modeling and verification," in *International Conference on Model and Data Engineering*, pp. 401–417, LNCS, Springer, 2018.

[51] L. Bellatreche, G. A. Chernishev, A. Corral, **Samir Ouchani**, and J. Vain, eds., *Advances in Model and Data Engineering in the Digitalization Era - MEDI 2021 International Workshops: DETECT, SIAS, CSMML, BIOC, HEDA, Tallinn, Estonia, June 21-23, 2021, Proceedings*, vol. 1481 of *Communications in Computer and Information Science*, Springer, 2021.

[52] **Samir Ouchani** and A. Khaled, "Security Assessment and Hardening of Autonomous Vehicles," in *The 15th International Conference on Risks and Security of Internet and Systems (CRISIS)*, LNCS Springer, 2020.

le cnam

Samir OUCHANI

Secure and Robust Cybe- Physical Systems

HESAM UNIVERSITÉ

Résumé : Les systèmes cyber-physiques intelligents (SCPS) sont des entités hétérogènes autonomes et interconnectées. Ils jouent un rôle central dans les infrastructures critiques. Ces systèmes sont devenus davantage dépendants des différentes technologies de communication embarquées. La pluralité de ces modules de communication augmente considérablement les possibilités d'attaques et rend les SCPS plus sensibles. En effet, ils peuvent engendrer de nouvelles catégories de vulnérabilités, ce qui pourrait entraîner des dommages importants. Mes travaux de recherche visent à développer des solutions garantissant les exigences fonctionnelles, la sécurité et la résilience des SCPS. Ma production scientifique s'appuie, entre autres, sur les méthodes formelles et les techniques de l'intelligence artificielle afin d'assurer la fiabilité de ces systèmes.
Mots clés : Sécurité, Fiabilité, Systèmes Cyber-Physiques, Fonctions physiques non clonables, Blockchain, Méthodes Formelles, Intélligence Artificielle.

Abstract: Smart Cyber-Physical Systems (SCPS) are heterogeneous inter-operable autonomous entities that play a crucial role in critical infrastructures. And as a result of supporting novel communication and remote control features, they became more dependent on connectivity, which increased attack surfaces and introduced errors that elevated the likelihood of SCPS errors. Additionally, they may cause new types of errors, faults, and vulnerabilities, leading to significant economic damage. In my ongoing and future research, I aim to develop scalable solutions that satisfy the systems' requirements, ensure security, and support resilience and synergy between the components of SCPS. My research activities involve the application of software engineering methodologies and artificial intelligence techniques to solve critical societal and industrial problems related to SCPS.

Keywords: Security, Reliability, Cyber-Physical System, Physical Unclonable Functions, Blockchains, Formal Methods, Artificial Intelligence.