Activity Report 2020-2021

Research-Team ArchWare

Software Architecture

Architecting Software-intensive Systems and Systems-of-Systems

D4 – Language and Software Engineering

Contents

This activity report covers the years of 2020 and 2021 of the ArchWare Research Team.

1 Team composition

Researchers and faculty

Flavio Oquendo, Full Professor/PR classe exceptionnelle, PEDR, Univ. Bretagne Sud (Head)
Isabelle Borne, Full Professor/PR 1ère classe, Univ. Bretagne Sud
Nicolas Belloir, Assistant Professor/MCF classe normale, Ecoles de St-Cyr
Jérémy Buisson, Associate Professor/MCF cl. normale, HDR, Ecoles de St-Cyr
Vanea Chiprianov, Assistant Professor/MCF cl. normale, Univ. Bretagne Sud (until June 2020)
Jamal El Hachem, Assistant Professor/MCF cl. normale, Univ. Bretagne Sud
Régis Fleurquin, Associate Professor/MCF hors cl., HDR, Univ. Bretagne Sud
Elena Leroux, Assistant Professor/MCF cl. normale, Univ. Bretagne Sud
Salah Sadou, Full Professor/PR 2ème classe, PEDR, Univ. Bretagne Sud

Associate members (IRISA/SEGULA Agreement)

Soraya Mesli-Kesraoui, PhD, SEGULA Technologies - R&D Division

Research engineers

Gersan Moguérou, Research Engineer (IGR 1ère classe), Université Bretagne Sud

PhD students

Raounak Benabidallah
Rymel Benabidallah
Nathalie Bouldoukian
Youcef Bouziane
Elyes Cherfa
Imane Cherfa
Mohamed Hicham Fendali
Elia Christy Fikany
Nan Zhang Messe
Paul Perrotin
Monica Buitrago Ramirez
Brendan Le Trionnaire
Thierry Waszak

External collaborators in particular in cotutelles of PhD theses

Thais Batista, Everton Cavalcante, Mohamed Ahmed Nacer, Djamel Meslati, Elisa Nakagawa, Jair Leite, Marcel Oliveira

Administrative assistant

Sylviane Boisadan, BIATSS, Université Bretagne Sud
Anne Idier, BIATSS, Université Bretagne Sud

2 Overall objectives

2.1 Overview

The ArchWare Research Team addresses the scientific and technological challenges raised by architecting complex software-intensive systems. Beyond static architectures (i.e. software architectures which are unchanging over time), we focus our research on the dynamic architectures of software-intensive systems (i.e. software architectures which change on the fly during run-time, according to the ways foreseen at design-time). Besides dynamic component-based and service-oriented systems, we address an emergent class of evolving software-intensive system that is increasingly shaping the future of our software-reliant world, the so-called System-of-Systems (SoS). SoSs exhibit evolutionary architectures (i.e. software architectures which change dynamically in ways not necessarily foreseen at design-time).

Indeed, since the dawn of computing, the complexity of software and the complexity of systems reliant on software have grown at a staggering rate. In particular, software-intensive systems have been rapidly evolved from being stand-alone systems in the past (often based on static architectures), to be part of networked systems in the present (often based on dynamic architectures), to increasingly become systems-of-systems in the coming future (based on dynamic, evolutionary architectures).

De facto, software-intensive systems have been independently developed, operated, managed, and evolved. Progressively, networks made communication and coordination possible among these autonomous systems, yielding a new kind of complex system, i.e. a system that is itself composed of systems. These systems of systems are evolutionary developed from systems to achieve missions not possible by each constituent system alone.

Different aspects of our lives and livelihoods have become overly dependent on some sort of software-intensive SoS. This is the case of SoSs found in different areas as diverse as aeronautics, automotive, energy, healthcare, manufacturing, and transportation; and applications that addresses societal needs as e.g. in environmental monitoring, distributed energy grids, emergency coordination, global traffic control, and smart cities.

Moreover, emergent platforms such as the Internet of Things and emergent classes of SoSs such as Cyber-Physical SoSs are accelerating the need of constructing rigorous foundations, languages, and tools for supporting the architecture and engineering of trustworthy SoSs.

Complexity is intrinsically associated to SoSs by its very nature that implies emergent behavior: in SoSs, missions are achieved through emergent behavior drawn from the interaction among constituent systems. Hence, complexity poses the need for separation of concerns between architecture and engineering: (i) architecture focuses on reasoning about interactions of parts and their emergent properties; (ii) engineering focuses on designing and constructing such parts and integrating them as architected.

Definitely, the software architecture forms the backbone for taming the complexity of trustworthy software-intensive systems, in particular in the case of evolving systems and systems-of-systems, where architecture descriptions provide the framework for designing, constructing, and dynamically evolving such complex systems, in particular

when they operate in unpredictable open-world environments.

Therefore, the endeavor of designing trustworthy systems evolved from architecting complicated systems in the last century, based on static architectures, to architecting trustworthy systems and systems-of-systems in this century, based on dynamic architectures. In particular, trustworthy SoSs, by their very nature, have intrinsic properties that are very hard to address.

Furthermore, the upcoming generation of trustworthy SoSs will operate in environments that are open in the sense of that they are only partially known at design-time. These open-world trustworthy SoSs, in opposite to current closed-world systems, run on pervasive devices and networks providing services that are dynamically selected and used to deliver more complex services, which themselves can be part of yet more complex services and so on. Furthermore, they will often operate in unpredictable environments.

Besides, in SoSs, architectures are designed to fulfill specified missions. Indeed, an important concern in the design of SoSs is the systematic modeling of both global and individual missions, as well as all relevant mission-related information. Missions play a key role in the SoS context since they define required capabilities of constituent systems and the interactions among these systems that lead to emergent behaviors towards the accomplishment of the global mission of the SoS.

Definitely, the unique characteristics of SoS raise a grand research challenge for the future of software-reliant systems in our industry and society due to its simultaneous intrinsic features, which are:

1. *Operational independence:* the participating systems not only can operate independently, they do operate independently. Hence, the challenge is to architect and engineer SoS in a way that enables its operations (acting to fulfill its own mission) without violating the independence of its constituent systems that are autonomous, acting to fulfill their own missions.

2. *Managerial independence:* the participating systems are managed independently, and may decide to evolve in ways that were not foreseen when they were originally composed. Hence, the challenge is to architect and engineer an SoS in a way that it is able to evolve itself to cope with independent decisions taken by the constituent systems and hence be able to continually fulfill its own mission.

3. *Distribution of constituent systems:* the participating systems are physically decoupled. Hence, the challenge is to architect and engineer the SoS in a way that matches the loose-coupled nature of these systems.

4. *Evolutionary development:* as a consequence of the independence of the constituent systems, an SoS as a whole may evolve over time to respond to changing characteristics of its environment, constituent systems or even of its own mission. Hence, the challenge is to architect and engineer SoS in a way that it is able to evolve itself to cope with these different kinds of evolution.

5. *Emergent behaviors:* from the collaboration of the participating systems may emerge new macroscale behaviors. Furthermore, these macroscale behaviors may be ephemeral because the systems composing the SoS evolve independently, which

may impact the availability of these behaviors. Hence, the challenge is to architect and engineer an SoS in a way that emergent behaviors and their subsequent evolution can be discovered and controlled.

In the case of an open-world environment, one can add the following characteristics:

1. *Unpredictable environment:* the environment in which the open-world SoS operates is only partially known at design-time, and thereby there will inevitably be novel situations to deal with at run-time. Hence, the challenge is to architect and engineer such a system in a way that it can dynamically accommodate to unprecedented situations while acting to fulfill its own mission.

2. *Unpredictable constituents:* the participating systems are only partially known at design-time. Hence, the challenge is to architect and engineer an open-world SoS in a way that constituent systems are dynamically selected, composed, operated, and evolved in a continuous way at run-time, in particular for achieving its own mission.

3. *Long-lasting:* as an open-world SoS is by nature a long-lasting system, re-architecting must be carried out dynamically. Hence, the challenge is to evolutionarily re-architects and evolves its construction without interrupting it.

The importance of developing novel theories and technologies for architecting and engineering SoSs is highlighted in several roadmaps targeting year 2020 and beyond.

In France, SoS architecture and engineering is explicitly targeted in the report prepared by the French Ministry of Economy as one of the key technologies for the period 2015-2025 (étude prospective sur les technologies clés 2015-2025, Direction Générale de la Compétitivité, de l'Industrie et des Services du Ministére de l'Economie).

In Europe, SoSs are explicitly targeted in the studies developed by the initiative of the European Commission, i.e. Directions in Systems-of-Systems Engineering, and different Networks of Excellence (NoE), in particular HiPEAC (NoE on high-performance and embedded computing systems) and HYCON2 (NoE on highly-complex and networked control systems) and different European Technological Platforms (ETP) and Industrial Associations (IA), specifically ARTEMIS-IA (industrial association for actors in embedded and cyber-physical systems) and NESSI-ETP (European platform on software and services) point out the relevance and timeliness of addressing the SoS challenge.

In 2014, two roadmaps for SoSs were produced under the support of the European Commission, issued from the CSAs ROAD2SoS (Development of strategic research and engineering roadmaps in Systems-of-Systems) and T-Area-SoS (Transatlantic research and education agenda in Systems-of-Systems). In 2015, the CSA CPSoS presented a research agenda for developing cyber-physical SoSs.

All these roadmaps show the importance of progressing from the current situation, where SoSs are basically developed in ad-hoc way in specific application sectors, to a scientific approach providing rigorous theories, technologies, and methodologies for mastering the complexity of SoSs in general (transversely to application domains).

It is worth to note that software-intensive system-of-systems is an emergent domain in the research community. The systematic mapping of the literature shows that 75% of the publications related to the architecture of systems-of-systems have been published in the last 5 years and 90% in the last 10 years. Furthermore, most of these publications raise open-issues after having experimented existing approaches for architecting systems-of-systems.

Overall, the long-term research challenge raised by SoSs calls for a novel paradigm and novel trustful approaches for architecting, analyzing, constructing, and assuring the continuous correctness of systems-of-systems, often deployed in unpredictable environments, taking into account all together their intrinsic characteristics.

The targeted breakthrough for the ArchWare Research Team is to conceive sound foundations and a novel holistic approach for architecting open-world trustworthy software-intensive SoSs, encompassing:

- Concepts and abstractions for formulating the architecture and re-architecture of SoS;
- Formalism and underlying computational model to rigorously specify the architecture and re-architecture of SoS;
- Mechanisms to construct, manage, and evolve SoSs driven by architecture descriptions, while resiliently enforcing their correctness, effectiveness, and efficiency;
- Concepts, formalisms and mechanisms for specifying and operating SoS missions, deriving abstract architectures, as well as generating concrete SoS architectures;
- Concepts, formalisms and mechanisms for specifying and enforcing safety/liveness properties as well as cybersecurity to achieve trustworthiness in SoS architectures.

Keywords: Software Architecture, Architecture Description, Architecture Analysis, Safety Architecture, Cybersecurity Architecture, Mission Specification, Software-intensive Systems, Software-intensive Systems-of-Systems, Architecture-based Evolutionary Development.

2.2 Scientific foundations

For addressing the scientific challenge raised for architecting SoS, the targeted breakthrough for the ArchWare Research Team is to conceive sound foundations and a novel holistic approach for architecting open-world critical software-intensive systems-of-systems, encompassing:

1. Architectural abstractions for formulating the architecture and re-architecture of SoS;
2. Formalism and underlying computational model to rigorously specify the architecture and re-architecture of SoS;

3. Mechanisms to construct, manage, and evolve SoSs driven by architecture descriptions, while resiliently enforcing their correctness, effectiveness, and efficiency;
4. Formalism and mechanisms for ensuring safety and cybersecurity at the architectural level and their transformations towards implementation.
5. Concepts and formalisms for specifying and operating SoS missions and generating abstract and concrete SoS architectures.

The research approach we adopt in the ArchWare Research Team for developing the expected breakthrough is based on well-principled design decisions:

1. To conceive architecture description, analysis, and evolution languages based on suitable SoS architectural abstractions;
2. To formally ground these SoS-specific architecture languages on well-established concurrent constraint process calculi and associated logics;
3. To conceptually and technologically ground the construction and management of SoSs on architecture descriptions defined by executable models;
4. To derive/generate abstract/concrete architectural descriptions from well-defined mission specifications.

2.3 Application domains

The ArchWare Research Team develops formalisms, languages and software technologies which are transverse to application domains while providing mechanisms for customization to different architectural styles and application areas. In these different application areas, extra-functional properties, in particular safety and cybersecurity, are addressed.

During 2020 and 2021, addressed applications areas include:

1. Internet-of-Things (IoT), Industrial Internet-of-Things (IIoT), Internet-of-Vehicles (IoV);
2. Intelligent Transportation Systems;
3. Battlefield Engineering;
4. Cybersecurity of major events reliant on digital systems.

3 Scientific achievements

3.1 Formal approaches for systems-of-systems architectures: the SoS Architecture Description Language (SosADL)

Keywords: Architecture Description Language (ADL), Mission Specification, Architecture Synthesis, Uncertainty, Fuzzy Architecture Description, Software-intensive Systems-of-Systems (SoS).

Participants: Flavio Oquendo, Thais Batista, Jérémy Buisson, Everton Cavalcante, Milena Guessi, Elisa Nakagawa, Jair Leite, Elena Leroux, Gersan Moguérou, Lidiane Santos, Eduardo Silva.

The architecture provides the right abstraction level to address the complexity of software-intensive Systems-of-Systems (SoSs). The research challenges raised by SoSs are fundamentally architectural: they are about how to organize the interactions among the constituent systems to enable the emergence of SoS-wide behaviors and properties derived from local behaviors and properties by acting only on their connections, without being able to act in the constituent systems themselves.

Formal architecture descriptions provide the framework for the design, construction, and dynamic evolution of SoSs.

From the architectural perspective, in single systems, the controlled characteristics of components under the authority of the system architect and the stable notion of connectors linking these components, mostly decided at design-time, is very different from the uncontrolled nature of constituent systems (the SoS architect has no or very limited authority on systems) and the role of connection among systems (in an SoS, connections among constituents are the main architectural elements for enabling emergent behavior to make possible to achieve the mission of an SoS).

The nature of systems architectures (in the sense of architectures of single systems) and systems-of-systems are very different:

- Systems architectures are described by extension. In the opposite, SoS architectures are described by intention.
- Systems architectures are described at design-time for developing the system based on design-time components. In the opposite, SoS architectures are defined at runtime for developing the SoS based on discovered constituents.
- Systems architectures often evolve offline. In the opposite, SoS architectures always evolves online.

In the sequel, the main results of this line of research produced in 2020 and 2021 are presented.

3.1.1 Empowering SosADL with mission description and verification

Keywords: Software Architecture, Systems-of-Systems, Mission Modeling, Formal Verification.

One of the prominent domains of nowadays software-intensive systems engineering concerns on designing and maintaining SoSs. Aiming to support the development of such kind of systems, mission models allow stakeholders to set objectives and resources for SoS, complementing the traditional requirements approach. In a mission model, designers specify the missions of the system and what the system is capable of doing to achieve them. On the other hand, one of the most desired features of any modeling

language is the possibility of formally verifying described properties. Enabling the stakeholders to check the degree of compliance of a model to a given set of properties fosters quality and simplifies the development process by shifting the problem resolution to the design level. The difficulties of SoS modeling also reflect on the verification processes: traditional verification methods were shown noneffective. In this context, this research track proposes a method to verify SoS models based on mission-related properties, using the mission modeling language mKAOS and the DynBLTL formalism. Our developed approach has being applied to SosADL. For details, see: [12].

3.1.2 Empowering SosADL with constraint-based architectural synthesis of concrete systems-of-systems

Keywords: Software Architecture, Architectural Synthesis, Constraints, Systems-of-Systems (SoS).

As SoSs leverage capabilities of heterogeneous systems for accomplishing complex combined behaviors, they pose new challenges to traditional software-intensive systems engineering practices that considered software architectures to be mostly static and stable. The software architecture of an SoS is inherently dynamic due to uncertainty surrounding its operational environment. While the abstract architecture offers a way to implicitly describe different forms taken by the software architecture at run time, it is still not sufficient to guarantee that all concrete architectures will automatically adhere to it. To address this issue, this research track developed a formal method, named Ark, supporting the architectural synthesis of SoSs. This is achieved by expressing abstract architectures as a set of constraints that must be valid for any concrete architecture of the abstractly defined SoS. This way, we can benefit from existing model-checking techniques to guarantee that all concrete architectures realized from such an abstract model will comply with well-formed rules. This method can be incorporated to a model-driven approach for bridging the gap between abstract and concrete architectural models. We validated the Ark method in a case study, showing how Ark can be used to support the synthesis of concrete architectures as well as the check of correctness and completeness of abstract architecture descriptions within the correct-by-construction approach. For details, see: [10].

3.1.3 Empowering SosADL with fuzzy architecture description for handling uncertainty in systems-of-systems

Keywords: Uncertainty, Fuzzy Theory, Architecture Description, System-of-Systems (SoS), Internet-of-Things (IoT), Self-Driving Vehicles Platooning.

Uncertainty is intrinsically associated with the architectural design of SoSs by its very nature, e.g. in Intelligent Transportation Systems (ITS). The consequent research question is thereby how to represent uncertainty in the description of an SoS architecture and subsequently use that representation to reason about SoS architectural properties. To address this research issue in the Internet-of-Things (IoT), this research track investigated the notion of epistemic uncertainty (i.e. uncertainty due to partial

knowledge) in the architectural design of IoT SoSs and proposes to enhance an existing SoS Architecture Description Language, named SosADL, with fuzzy concepts and constructs underlain by Fuzzy Theory, demonstrating its effectiveness in SoS platooning architectures of self-driving vehicles. This research track also addressed the validation of SosADL via a case study of the Fuzzy SosADL, supported by a controlled experiment, on the description of SoS architectures (in terms of fuzzy models) for platooning of self-driving vehicles, a typical application of intelligent transportation where a group of autonomous vehicles, subject to uncertainties, travels together in road convoys. For details, see: [22] and [23].

3.2 Formal approaches for systems architectures: the Systems Architecture Description Language (SysADL)

Keywords: Architecture Description Language (ADL), Architecture Modeling, Executable Architecture Specifications, Verifiable Architecture Specifications, Software-intensive Systems (Sys).

Participants: Flavio Oquendo, Camila Araujo, Thais Batista, Everton Cavalcante, Fagner Dias, Jair Leite, Marcel Oliveira, Lidiane Santos, Eduardo Silva.

The architecture provides the right abstraction level to address the complexity of Software-intensive Systems (Sys). This research line addresses the research challenges raised by architecture modeling based on the ISO SysML notation. We defined an Architecture Description Language, named SysADL, as a specialization of the ISO-OMG SysML standard to software architecture description. SysADL brings together the expressive power of software architecture description languages (ADLs) for architecture description, with a standard language used by the industry (SysML). SysADL defines viewpoints to describe the structure, the behavior, and the execution of a software architecture of a software intensive-system.

From the architectural perspective, in single systems, the controlled characteristics of components under the authority of the system architect and the stable notion of connectors linking these components, is mostly decided at design-time. The main research challenge is thereby to enforce correctness-by-design associating on the one hand a software architect-friendly notation with on the other hand formal representations for supporting formal refinement and analysis.

In the sequel, the main results of this line of research produced in 2020 and 2021 are presented.

3.2.1 Formalizing SysADL: model-driven approach for generating formal software architecture descriptions: from SysADL to π-ADL

Keywords: Model-Driven Development, Model Transformation, Architecture Description Language, Formal Verification, SysML.

The critical nature of many complex software-intensive systems requires formal ar-

chitecture descriptions for supporting automated architectural analysis regarding correctness properties. Due to the challenges of adopting formal approaches, many architects have preferred using standard notations, in particular SysML and their derivatives, to describe the structure and behavior of software architectures of software-intensive systems. However, SysML and other semi-formal notations have limitations regarding the sought support for architectural analysis. This research track developed an approach to bridge the rigor of formal architecture descriptions and the ease of use of SysML-based notations widely used elsewhere. The main concern is providing formal semantics to SysADL, a SysML-based language to describe software-intensive system architectures. The formal semantics is provided by π-ADL, a formal architecture description language. A model-to-model transformation was defined and implemented to concretize the mapping between the elements of these languages and hence automatically generate formal architecture descriptions in π-ADL from SysADL. This work developed a proof-of-concept to validate the mapping between SysADL and π-ADL and an exploratory study on the transformation performance. For details, see: [13].

3.2.2 Empowering SysADL with formal verification: from SysADL to CSP

Keywords: Software Architecture Description, Formal Verification, Correctness Properties, CSP, SysML.

One of the many purposes of software architecture descriptions is contributing to an early analysis of the architecture with respect to quality attributes. The critical nature of many software systems calls for formal approaches aiming at precisely verifying if their designed architectures can meet important properties such as consistency, completeness, and correctness. In this context, it is worthwhile investigating the role of architecture descriptions to support the formal verification of software architectures to ensure their quality, as well as how such a process happens and is supported by existing languages and verification tools. To evaluate the research landscape on this subject, we have carried out in this reaserch track a systematic mapping study in which we collected and analyzed studies available in the literature on formal verification of architecture descriptions. This research track contributed with a structured overview and taxonomy of the current state of the art on this topic as well as the elicitation of key open questions. For details, see: [18].

3.2.3 Enhancing SysADL with architectural styles: architectural style for IoT systems

Keywords: Architecture Description, Architectural Style, Internet-of-Things (IoT).

In recent years, we have witnessed the emergence of the Internet of Things (IoT) paradigm as means of supporting connectivity, interaction, and integration of "things", which collaborate with each other to achieve common goals. The characteristics of IoT have challenged software engineering, including how to successfully architect complex systems while meeting business goals and satisfying important quality attributes. The

use of software architectural styles provides a solid foundation for mitigating risks arisen from the complexity of these systems. In this research track, we developed an architectural style for IoT systems in conformance with the IoT conceptual model defined by the ISO/IEC 30141 International Standard. We specify the style by using the SysADL language in terms of the description of its architectural elements, structure, behavior, and constraints. We validated the style through the architectural design of a real-world IoT system and evaluated its use through an exploratory experiment. For details, see: [24] and [25].

3.3 Addressing design and operational aspects of software-intensive systems-of-systems

Keywords: Software Architecture, Modeling, Simulation, Reconfiguration, Systems-of-Systems.

Participants: Salah Sadou, Rymel Benabidallah, Isabelle Borne, Jérémy Buisson, Armel Esnault, Régis Fleurquin, Mohamed Ahmed Nacer, Franck Petitdemange.

The architecture provides the right abstraction level to address the complexity of SoSs. The research challenges raised by SoSs are fundamentally architectural: they are about how to organize the interactions among the constituent systems to enable the emergence of SoS-wide behaviors and properties derived from local behaviors and properties by acting only on their connections, without being able to act in the constituent systems themselves.

In the architectural design phase as well as in the operational phase, different techniques need to be applied to validate the conceived architectures, i.e. simulation, reconfiguration, evoloution.

In the sequel, the main results of this line of research produced in 2020 and 2021 are presented.

3.3.1 SoS simulation: simulating systems-of-systems using situation/reaction paradigm

Keywords: Simulation, Situation-Reaction Paradigm, Systems-of-Systems.

Modeling and simulation play a major role in complex systems engineering. In SoS engineering, a special case of complex systems engineering, they help to better understand and identify the side effects associated with the integration of autonomous constituent systems. Simulation is also a way of apprehending the emerging behaviors from this integration. The dynamic and evolving nature of the SoS environment has led us to rely on their most stable part to define them, namely their mission. In this reaserch track, we developed a simulation framework for SoS based on a mission conceptual model. Mission is defined as a set of situations that require reactions. Situations are defined by rules on facts related to the SoS environment. Reactions are defined as

orchestrations of services from constituent systems (subsystems) that must be triggered when a situation is identified. We validated this simulation approach through a case study on a Telediabet SoS. For details, see: [7] and [3].

3.3.2 SoS reconfiguration: defining a design process for system-of-systems reconfigurations

Keywords: Dynamic Reconfiguration, System-of-Systems.

Dynamic reconfiguration has been studied mainly in the context of distributed systems based on components. Whereas in the case of distributed systems, the focus is on autonomous reaction to environmental changes and automatic self-adaptation, the evolutionary development of an SoS turns reconfiguration into an engineering activity. When a constituent system decides to quit, the SoS may react and reconfigure itself automatically, but when a new constituent system must be recruited, the architect of the SoS may have to sign contracts for, for instance, financial or legal reasons. Moreover, a new constituent system may even have to be designed and manufactured. Moreover, the SoS is also reconfigured when new requirements arise, resulting in the need for a new architecture. To address these issues, in this research track we view reconfiguration as the design, implementation, and deployment of any change in the SoS architecture. In this research track, we developed a design process to guide the SoS architect in charge of designing a reconfiguration, noting the tasks that must be accomplished to develop a reconfiguration for an evolutionary development. We apply our process to one of the configurations obtained from different case study scenarios. For details, see: [11].

3.3.3 SoS evolution: characterizing the evolution of systems-of-systems

Keywords: Software Evolution, Systems-of-Systems.

This research track aims to characterize the circumstances of evolution in SoSs. First an analysis of the nature of evolution in SoSs, comparing it with that of ordinary systems, was conducted. Through this analysis, we put evolution of SoSs in comparison with that of ordinary systems and attempt to identify the aspects where they differ or coincide. We then further attempted to provide actionable strategies for effectively managing SoS evolution. Special focus of this research track is made on the aspects related to why and how evolution occurs for SoS and how it can be controlled.

3.3.4 SoS for the battlefield

Keywords: Military SoS, Battlefield Engineering, Model-Based Engineering, Operation Orders, Systems-of-Systems.

Digitalization of the whole society will change the way SoSs have to be considered. Remaining independently operated and managed, SoSs increase their collaboration skills using shared or cooperated information systems. People can be seen as particular

digital subsystems due to smart equipment they can use. Military operations, which are considered as typical SoS, are no exception to this fact. Foreseen evolution with the current French programs is towards mixing command post, human soldiers, their equipment, remotely operated robots and autonomous robots interacting in a single SoS. New operational doctrines have to be created to take advantage of those new capabilities. In this research track, we developed new methods supported by tools inspired by software engineering to create new automated capabilities in battlefield engineering. They support the direction which should be considered in the area of battlefield engineering in order to deal with those new capabilities. Inspired from Model-Based Engineering, we developed a metamodel inspired by the French PROTERRE doctrine (soldier to platoon levels) in such a way that the metamodel is suitable in a multimodel interaction scenario, including both human and robotic systems.

3.4 Addressing cybersecurity in software-intensive systems and systems-of-systems

Keywords: Security by Design, Software Architecture, Vulnerability Modeling, Attack Modeling, Threat Modeling.

Participants: Nicolas Belloir, Raounak Benabidallah, Isabelle Borne, Jérémy Buisson, Vanea Chiprianov, Régis Fleurquin, Monica Buitrago, Salah Sadou.

The architecture provides the right abstraction level to address the issue of cybersecurity-by-design in software-intensive systems and SoSs.

This research line addresses the research challenges raised by architecting secure software-intensive systems and SoSs. Different aspects of cybersecurity have been addressed at different levels, directly or indirectly related with the systems' software architecture.

In the sequel, the main results of this line of research produced in 2020 and 2021 are presented.

3.4.1 Security by design: asset-based approach to bridge the gap between software architects and security experts

Keywords: Security by Design, Software Architecture, Security Assistance, Attack Patterns, Threat Modeling, Asset-based Reference model, Asset Identification.

Security must be taken into account at all stages of the software development life cycle. One of the key approaches for addressing this issue is threat modelling, which involves the collaboration of several actors, in particular software architects and security experts, during brainstorming sessions. If architects are familiar with the design artifacts they handle as well as have the right skills in assembly practices, they are not always well-trained in cybersecurity. For improving threat modeling approaches, and more generally the considerations of cybersecurity concerns at the architectural design stage, we defined in this research track a security assistance based on the concept of

"assets". This assistance is dedicated to architects and helps them to design secure systems, even if they have only limited security knowledge. This assistance alerts them, during their modelling activity, by displaying the possible threats associated to the system being designed, and may suggest a list of possible control mechanisms to mitigate the vulnerable parts of this system at the architectural design stage. The proposed assistance is also based on a security-oriented knowledge base built upon international references such as CAPEC and CWE. The process enables the identification of both relevant and vulnerable assets, which facilitates threat enumeration during the brainstorming sessions. We applied the proposed process with a case study and show its usefulness in facilitating threat enumeration and improving the existing threat modelling processes. For details, see: [4], [20], and [21].

3.4.2 Security vulnerabilities: automatic identification of security vulnerabilities in software systems

Keywords: Vulnerability Identification, Static Analysis, Prediction Models.

The threat caused by software vulnerabilities is growing exponentially. This phenomenon is due, on the one hand, to the omnipresence of software, and on the other hand, to the large number of existing vulnerabilities. To deal with this issue, several strategies have been developed over time. Some of them aim to establish good development practices and integrate them right from the design phase, while others consist of carrying out security inspections by identifying vulnerable areas. This research track is related to the second category of work and focuses on the construction of vulnerability prediction models. The first outcome of this research track focuses on the corpus construction. It is based on the design of vulnerability meta-scanners allowing to identify code vulnerabilities efficiently. This consists in combining several static analysis tools based on their individual performance for each category of vulnerabilities. The second outcome corresponds to the SecureQualitas corpus which consists of a corpus of Java applications annotated with the vulnerabilities they contain. We build this corpus using a meta-scanner built with three vulnerability analysis tools. The third outcome is to build a prediction model of vulnerable code. We opted and studied the use of quality metrics to characterize code and we have studied the performance of the models both on categories of vulnerabilities learned by the models and on categories not yet known by the model. The results of our experiments showed the efficiency of the models on both populations of vulnerabilities: known and unknown. For details, see: [2].

3.4.3 Security measures: measuring security in architecture descriptions

Keywords: Software Architecture Description, Cybersecurity, Patterns, Metrics.

When designing a software architecture, some well-established patterns are known to improve the security of the software system being designed. For instance, having a well-identified single entry point to a subsystem eases the interactions with the policy enforcement point and policy decision point that implement access control to the services and resources of that subsystem. But whether such patterns are actually and

suitably used in software design has still to be measured. Our first outcome to this issue is a literature review. It confirmed that the architecture design affects the security hardening. We found several metrics that attempt to assess the susceptibility of vulnerabilities at the code level. But, our literature review also shows the lack of existing metrics at the architecture level to address the specific question of design-time security.

4 Software development

4.1 The SoS architect studio for SosADL: SosADL Studio

Participants: Gersan Moguérou, Jérémy Buisson, Milena Guessi, Elena Leroux, Valdemar Neto, Flavio Oquendo.

SosADL Studio, the SosADL Architecture Development Environment, is a novel environment for description, verification, simulation, and compilation/execution of SoS architectures. With SosADL Studio, SoS architectures are described using SosADL, an Architecture Description Language based on process algebra with concurrent constraints, and on a meta-model defining SoS concepts. Because constituents of an SoS are not known at design time, SosADL promotes a declarative approach of architecture families. At runtime, the SoS evolves within such a family depending on the discovery of concrete constituents. In particular, SosADL Studio enables to guarantee the correctness of SoS architectures.

At the end of 2019, the SosADL Studio included the following modules in its alpha version.

4.1.1 The type system in Coq, the type-checker and the proof generator

Participants: Jérémy BuissonGersan Moguérou.

The type-checker is based on the SosADL type system written in Coq, which covers 2/3 of the SoSADL language. Coq proofs are generated after each successful type checking, enabling the verification of the type-checker according to the type system.

4.1.2 SosADL2Alloy: generating concrete SoS architectures based on SosADL

Participants: Milena Guessi, Gersan Moguérou, Flavio Oquendo,.

The concrete architecture generator (SosADL2Alloy) module automatically transforms a SosADL abstract architecture into an abstract architecture in Alloy, and generates a Java class to launch a SAT solver through the Alloy Analyzer. The SAT solutions represent SosADL concrete architectures. During the integration of this module into the SosADL Studio, it has been improved to represent the generated concrete architectures in SosADL.

4.1.3 SosADL2DEVS: generating and simulating concrete architectures

Participants: Valdemar Neto, Wallace Manzano, Gersan Moguérou.

The SosADL2DEVS generator takes one concrete architecture as input and generates a DEVS program, which can be verified using ioSTS/Uppaal and simulated using the MS4ME simulation tool. The simulations generate traces. A client-server link between MS4ME and PlasmaLab enables Statistical Model Checking, by reusing traces of the simulation. The SosADL2DEVS module now generates DEVS programs, which can evolve dynamically during a simulation inside MS4ME.

This module has been integrated in the SosADL Studio. Currently, this module translates SosADL concrete architectures into DEVSNL, the language supported by MS4ME. This dependence with MS4ME requires running the simulation on Windows or Linux.

4.1.4 SosADL2IoSTS: the SosADL support for architecture verification

Participants: Elena Leroux, Gersan Moguérou.

The SosADL2IoSTS generator takes one concrete architecture, and generates an ioSTS model in order to verify functional properties of SoS. The development of the translator from ioSTS to Uppaal is partially terminated.

4.1.5 The SoSADL Studio IDE

Participants: Gersan Moguérou, Jérémy Buisson, Elena Leroux, Milena Guessi, Valdemar Neto, Flavio Oquendo.

The SoSADL Studio provides an Integrated Development Environment (IDE), a simulator, a model-checker, and a statistical model-checker.

The SosADL Studio was developed under Xtext/Eclipse. It integrates the above modules into an IDE, which provides a syntactical editor to define an abstract SoS architecture, and then enable the following workflow:

- The type-checker validates the abstract SoS architecture written in SosADL, and generate a Coq proof. This proof can be verified using the Coq proof assistant, according to the SosADL type system written in Coq.
- The concrete SoS architectures are then generated, by the execution of the SosADL2Alloy module.
- Each concrete architecture is transformed into ioSTS, and then into an Uppaal NTA, in order to verify functional properties by Model Checking.
- Each concrete architecture is transformed into a DEVS program, by the execution of the SosADL2DEVS module, and simulated using the MS4ME tool. The traces of the simulation enable Statistical Model Checking in PlasmaLab.

Based on all these modules and supported execution steps of the SosADL workflow, the overall SosADL Studio was implemented. We produced the SosADL eclipse plugin in its alpha version.

The development of the beta version of the SosADL Studio started in 2020. The grammar and meta-model have been completely redesigned using Eclipse/Xtext for simplifying its maintenance. The scoping and type-checker have been rewritten following Xtext principles. Generating Coq proofs have not yet been integrated. At the end of 2021, the generation of concrete architectures was near to be terminated, finalizing the core of the IDE.

5 Contracts and collaborations

5.1 National initiatives

- Public-private collaboration on the cybersecurity of large public events between the UniversitBretagne Sud, ENGIE and other companies in the domain of the cybersecurity of systems-of-systems. This ongoing collaboration aims to support the design and operation of large-scale sociotechnical systems-of-systems in open environments. It, in particular, aims to support the 2024 Summer Olympics in Paris. The ARCHWARE team brings to this joint R&D project its expertise on security-by-design for mastering emergent behaviors in sociotechnical SoS architectures.

5.2 Bilateral industry grants

- SEGULA Engineering (Numéro de contrat Ouest Valorisation 2018-01307): Bilateral collaboration on cybersecurity in software architecture for industrial systems and systems-of-systems in the industrial internet-of-things: Bilateral collaboration between ARCHWARE and the R&D division of SEGULA, a multinational systems engineering company developing large-scale systems and SoSs in different domains, including automotive, aeronautics, naval, and railway engineering. This ongoing collaboration aims to support the model-based engineering of critical systems-of-systems in the industrial internet of things. The ARCHWARE team brings to this joint R&D project its expertise on formal approaches for the specification and verification of software architectures of SoSs.

5.3 Collaborations

National research networks:

- Members of ARCHWARE actively participates in the GDR GPL (Groupement de Recherche Génie de la Programmation et du Logiciel - INS2I-CNRS), in particular the team will organize the next Journées nationales du GDR GPL in Vannes (initially planned for June 2021 and then reported due to the Covid pandemic to June 2022).

International research networks:

- Flavio Oquendo has a collaboration with PRAIA.AI - Applied Research Centers in Artificial Intelligence, Brazil, as part of the PRAIA International Committee.

National collaborations with joint publications:

- Flavio Oquendo has a collaboration on systems-of-systems with Khalil Drira (LAAS-CNRS);
- Salah Sadou has a collaboration on reuse of architectural constraints with Chouki Tibermancine (LIRMM);

International collaborations with joint publications:

- Flavio Oquendo has a collaboration on software architecture with Thais Batista (UFRN - Federal University of Rio Grande do Norte, Natal, Brazil);
- Flavio Oquendo has a collaboration on systems-of-systems with Elisa Nakagawa (USP - University of Sao Paulo - ICMC Research Institute, Sao Carlos, Brazil);
- Jamal El Hachem has a collaboration on systems-of-systems security with Ali Babar (University of Adelaide, Adelaide, Australia).

National collaborations with joint PhD supervision:

- Jérémy Buisson:
  - École de l'Air, Salon-de-Provence, France.
- Salah Sadou:
  - University of Science and Technology of Houari Boumedienne, Alger, Algeria (Mohamed Ahmed Nacer).

International Collaborations with joint PhD supervision:

Isabelle Borne:

- University of Badji Mokhtar, Annaba, Algeria (Djamel Meslati).
- Flavio Oquendo:
  - UFRN - Federal University of Rio Grande do Norte, Natal, Brazil (Thais Batista);
  - USP - University of Sao Paulo - ICMC Research Institute, Sao Carlos, Brazil (Elisa Nakagawa);
  - Fraunhofer IESE - Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany (Pablo O. Antonino, in preparation).
- Salah Sadou:
  - University of Science and Technology of Houari Boumedienne, Alger, Algeria (Mohamed Ahmed Nacer).

6 Dissemination

6.1 Promoting scientific activities

Research and Doctoral Supervising Awards (PEDR)

- Flavio Oquendo: PEDR (2020-2024)
- Salah Sadou: PEDR (2018-2022)

Chair/member of conference steering committees

- Flavio Oquendo:
  - European Conference on Software Architecture - ECSA (Steering Committee Chair);
  - IEEE International Conference on Software Architecture - ICSA (Steering Committee Member);
  - Conférence francophone sur les architectures logicielles - CAL (Steering Committee Member);
  - ACM International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems (technically co-sponsored by ACM SIGSOFT and ACM SIGPLAN) - SESOS (Steering Committee Member).
- Salah Sadou:
  - CIEL: French Conference on Software Engineering (Steering Committee Chair).

Chair/member of conference program committees

- Isabelle Borne:
  - SOSE: IEEE International Conference on System-of-Systems Engineering, 2020 and 2021;
  - AROSA: IEEE WETICE Conference Track on Adaptive and Reconfigurable Service-oriented and component-based Applications and Architectures, 2020 and 2021;
  - SESOS: ACM/IEEE ICSE International Workshop on Software Engineering for Systems-of-Systems and Ecosystems, 2020 and 2021;
  - ASOCA: Workshop on Adaptive Service-oriented and Cloud Applications), 2020 and 2021;
  - IWST: International Workshop on Smalltalk Technologies, 2020.
- Jérémy Buisson:
  - ICCS: International Conference on Computational Science, 2020 and 2021.

- Flavio Oquendo:
  - ICSA: IEEE International Conference on Software Architecture, 2020 and 2021;
  - ECSA: European Conference on Software Architecture, 2020 and 2021;
  - SOSE: IEEE International Conference on System-of-Systems Engineering, 2020 and 2021;
  - ICT: International Conference on Telecommunications, 2020 and 2021;
  - ICWNES: International Conference on Wireless Networks and Embedded Systems, 2021;
  - CYBER: International Conference on Cyber-Technologies and Cyber-Systems, 2020;
  - CPSIOT: International Conference on Cyber Physical Systems and IoT, 2020 and 2021;
  - ICSOFT: International Conference on Software and Data Technologies, 2020;
  - ICSEA: International Conference on Software Engineering Advances, 2020 and 2021;
  - ICAS: International Conference on Autonomic and Autonomous Systems, 2020 and 2021;
  - SESOS: ACM/IEEE ICSE International Workshop on Software Engineering for Systems-of-Systems and Ecosystems, 2020 and 2021;
  - COMPLEXIS: International Conference on Complexity, 2020 and 2021;
  - CoDIT: International Conference on Control, Decision and Information Technologies, 2020;
  - SSCC: Symposium on Solutions for Smart Cities Challenges, 2020;
  - SBES: Brazilian Symposium on Software Engineering, 2020;
  - AROSA: IEEE WETICE Conference Track on Adaptive and Reconfigurable Service-oriented and component-based Applications and Architectures, 2020 and 2021;
  - SAC-SiSoS: Software-intensive Systems-of-Systems Track of the ACM SIGAPP Symposium On Applied Computing, 2020 and 2021;
  - ICSE-JSEET: International Conference on Software Engineering - Joint Track on Software Engineering Education and Training, 2020 and 2021;
  - QUATIC-SQET: Software Quality Education and Training, 2020;
  - ICSA-NEMI: ICSA New and Emerging Ideas, 2020;
  - DE&I: ECSA Diversity, Equity and Inclusion, 2021.

6.1.1 International invited talks

Webinars

- SoSECIE: Systems of Systems Engineering Collaborators Information Exchange Webinar (MITRE and USA National Defense Industrial Association's SoS Engineering Committee) on Fuzzy Architecture Description for Handling Uncertainty in Systems-of-Systems in the Internet-of-Things, 2021.

6.1.2 International journal boards

Member of the Editorial Boards

- Flavio Oquendo:
  - Springer Journal of Software Engineering Research and Development (Member of the Editorial Board).

6.1.3 Scientific expertise

- Flavio Oquendo:
  - Scientific Expert acting as reviewer and evaluator of R&D Projects for the European Commission (Horizon H2020 and Horizon Europe);
  - Scientific Expert acting as evaluator of R&D Proposals for the ANR (Agence Nationale de la Recherche) on Software Sciences and Engineering;
  - Scientific Expert acting as evaluator of R&D Projects for the MESRI - CIR on Software Engineering and Technologies;
  - Scientific Expert acting as evaluator of R&D Proposals for the CZSF (Czech Science Foundation) on Software Sciences and Engineering (Czech Republic);
  - Scientific Expert acting as evaluator of R&D Proposals for the Helmholtz Association of German Research Centers on Software Sciences and Engineering (Germany);
  - Scientific Expert acting as evaluator of R&D Proposals for the FWO (Research Foundation Flanders) on Software Sciences and Engineering, including the SBO Strategic Basic Research Program (Belgium);
  - Scientific Expert acting as evaluator of R&D Proposals for the ESF European Science Foundation on Software Sciences and Engineering (Belgium);
  - Distinguished Professor acting as evaluator of Scientific Standing and Achievements to Full Professorship for different universities in Europe on Computer Sciences and Software Engineering.

6.1.4 Laboratory administration

- Isabelle Borne: Responsible of the Site of Vannes for IRISA (until December 2020).

6.1.5 Academic council

- Salah Sadou: Member of the CAC (Commission recherche du conseil académique) of Université Bretagne Sud.

6.2 Teaching

6.2.1 University teaching

- Academics of ARCHWARE teach at the Research Master on Computer Science of Université Bretagne Sud which is part of the regional SIF master administered by a consortium of the main computer science universities and graduate schools in Brittany: Université de Rennes 1, Université de Bretagne Sud, ENS Rennes, National Institute of Applied Sciences, Rennes (INSA) and CentraleSupélec.

6.2.2 Teaching responsibility

- Jérémy Buisson: Deputy head of the Mastère Spécialisé "Conduite des Opérations et Gestion des Crises Cyber" of Académie Militaire de Saint-Cyr Coëtquidan;
- Elena Leroux: Study Director of the ENSIBS School of Engineering;
- Flavio Oquendo: Head of the Research Master Degree on Computing of Université Bretagne Sud (part of the regional SIF research master in Computer Science headed by Université Rennes 1);
- Salah Sadou: Head of the Engineering Degree on Software Cybersecurity of ENSIBS School of Engineering.

Elena Leroux: Study Director of the Engineering Degrees of the ENSIBS School of Engineering;

Doctoral dissertations and "Habilitation" theses

[1] D. Beaulaton, *Security Analysis of IoT Systems using Attack Trees*, Theses, Université de Bretagne Sud, December 2019, `https://tel.archives-ouvertes.fr/tel-02893847`.

[2] R. Benabidallah, *Identification automatique des vulnérabilités de sécurité dans les systèmes logiciels*, Theses, Université de Bretagne Sud, November 2020, `https://tel.archives-ouvertes.fr/tel-03326519`.

[3] R. Benabidallah, *Approche de simulation de systèmes de systèmes pour l'identification de comportements émergents*, Theses, USTHB & Université de Bretagne Sud, April 2021.

[4] N. Z. Messe, *Security by Design : An asset-based approach to bridge the gap between architects and security experts*, Theses, Université de Bretagne Sud, January 2021, `https://tel.archives-ouvertes.fr/tel-03407189`.

[5] F. Petitdemange, *Développement évolutionnaire de systèmes de systèmes avec une approche par patron de reconfiguration dynamique*, Theses, Université de Bretagne Sud, December 2018, `https://tel.archives-ouvertes.fr/tel-02073913`.

Articles in referred journals and book chapters

[6] N. Belloir, J. Buisson, L. Touseau, "Model-Driven Engineering as the Interface for Tactical Operation Order of Mixed Robot/Human Platoons", *in : Developments and Advances in Defense and Security*, *Smart Innovation, Systems and Technologies*, *255*, Springer Singapore, October 2022, p. 205–214, https://hal.archives-ouvertes.fr/hal-03418759.

[7] R. Benabidallah, S. Sadou, A. Esnault, M. A. Nacer, "Simulating systems of systems using situation/reaction paradigm", *Concurrency and Computation: Practice and Experience 32*, 15, August 2020, https://hal.archives-ouvertes.fr/hal-02896768.

[8] Y. Bouziane, M. K. Abdi, S. Sadou, "Automatically Labelled Software Topic Model", *International journal of open source software processes 11*, 1, January 2020, p. 57–78, https://hal.archives-ouvertes.fr/hal-02896775.

[9] L. Garcés, F. Oquendo, E. Y. Nakagawa, "Assessment of Reference Architectures and Reference Models for Ambient Assisted Living Systems", *International Journal of E-Health and Medical Communications (IJEHMC) 11*, 1, January 2020, p. 17–36, https://hal.archives-ouvertes.fr/hal-02569215.

[10] M. Guessi, F. Oquendo, E. Y. Nakagawa, "Ark: a constraint-based method for architectural synthesis of smart systems", *Software and Systems Modeling 19*, 3, May 2020, p. 741–762, https://hal.archives-ouvertes.fr/hal-02570176.

[11] F. Petitdemange, I. Borne, J. Buisson, "Design Process for System of Systems Reconfigurations", *Systems Engineering 24*, 2, March 2021, p. 69–82, https://hal.archives-ouvertes.fr/hal-03161725.

[12] E. Silva, T. Batista, F. Oquendo, "On the verification of mission-related properties in software-intensive systems-of-systems architectural design", *Science of Computer Programming 192*, June 2020, p. 102425, https://hal.archives-ouvertes.fr/hal-02570170.

Publications in Conferences and Workshops

[13] C. Araújo, T. Batista, E. Cavalcante, F. Oquendo, "Generating Formal Software Architecture Descriptions from Semi-Formal SysML-Based Models: A Model-Driven Approach", *in : ICCSA 2021 - 21st International Conference on Computational Science and Its Applications*, Cagliari, Italy, September 2021, https://hal.archives-ouvertes.fr/hal-03584963.

[14] J. Buisson, J. L. M. M, N. Belloir, "Digitalization in Next Generation C2: Research Agenda from Model-Based Engineering Perspective", *in : 2020 IEEE 15th International Conference of System of Systems Engineering (SoSE)*, IEEE, p. 000243–000248, Budapest, France, June 2020, https://hal.archives-ouvertes.fr/hal-02887623.

[15] E. Cherfa, S. M. Kesraoui, C. Tibermacine, R. Fleurquin, S. Sadou, "On investigating Metamodel Inaccurate Structures", *in : SAC 2020 - 35th Annual ACM Symposium on Applied Computing*, p. 1642–1649, Brno, Czech Republic, March 2020, https://hal-lirmm.ccsd.cnrs.fr/lirmm-03290255.

[16] E. Cherfa, S. Mesli-Kesraoui, C. Tibermacine, R. Fleurquin, S. Sadou, "Identifying Metamodel Inaccurate Structures During Metamodel/Constraint Co-Evolution",

in : MODELS 2021 - ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems, IEEE, p. 24–34, Fukuoka, Japan, October 2021, https://hal-lirmm.ccsd.cnrs.fr/lirmm-03521022.

[17] A. Delecolle, R. S. Lima, V. V. G. Neto, J. Buisson, "Architectural Strategy to Enhance the Availability Quality Attribute in System-of-Systems Architectures: a Case Study", *in : 2020 IEEE 15th International Conference of System of Systems Engineering (SoSE)*, IEEE, p. 93–98, Budapest, France, June 2020, https://hal.archives-ouvertes.fr/hal-02887620.

[18] F. M. Dias, M. Oliveira, T. Batista, E. Cavalcante, J. Leite, C. Araújo, F. Oquendo, "Empowering SysML-Based Software Architecture Description with Formal Verification: From SysADL to CSP", *in : 14th European Conference on Software Architecture (ECSA)*, L'Aquila, Italy, September 2020, https://hal.archives-ouvertes.fr/hal-03585038.

[19] S. Mesli-Kesraoui, O. Goubali, D. Kesraoui, I. Eloumami, F. Oquendo, "Formal Verification of the Race Condition Vulnerability in Ladder Programs", *in : 2020 IEEE Conference on Control Technology and Applications (CCTA)*, IEEE, p. 892–897, Montreal, Canada, August 2020, https://hal.archives-ouvertes.fr/hal-03585041.

[20] N. Messe, N. Belloir, V. Chiprianov, J. El-Hachem, R. Fleurquin, S. Sadou, "An Asset-Based Assistance for Secure by Design", *in : APSEC 2020 - 27th Asia-Pacific Software Engineering Conference*, p. 1–10, Singapore, Singapore, December 2020, https://hal.archives-ouvertes.fr/hal-02990897.

[21] N. Messe, V. Chiprianov, N. Belloir, J. El-Hachem, R. Fleurquin, S. Sadou, "Asset-Oriented Threat Modeling", *in : TrustCom 2020 - 19th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, IEEE, p. 1–11, Guangzhou, China, December 2020, https://hal.archives-ouvertes.fr/hal-02990919.

[22] F. Oquendo, "Fuzzy Architecture Description for Handling Uncertainty in IoT Systems-of-Systems", *in : 2020 IEEE 15th International Conference of System of Systems Engineering (SoSE)*, IEEE, p. 555–562, Budapest, Hungary, June 2020, https://hal.archives-ouvertes.fr/hal-03585030.

[23] F. Oquendo, "Case Study on the Fuzzy Architecture Description of Cyber-Physical SoS under Uncertainty", *in : 2021 16th International Conference of System of Systems Engineering (SoSE)*, IEEE, p. 61–68, Västerås, Sweden, June 2021, https://hal.archives-ouvertes.fr/hal-03585047.

[24] L. Santos, E. Silva, T. Batista, E. Cavalcante, J. Leite, F. Oquendo, "An Architectural Style for Internet of Things Systems", *in : The 35th ACM/SIGAPP Symposium on Applied Computing (SAC)*, Brno, Czech Republic, March 2020, https://hal.archives-ouvertes.fr/hal-02570200.

[25] L. Santos, E. Silva, T. Batista, J. Leite, E. Cavalcante, F. Oquendo, "Evaluating a SysML-based Graphical Notation for Modeling Internet of Things System Architectures", *in : 2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*, IEEE, p. 1–6, New Orleans, United States, June 2020, https://hal.archives-ouvertes.fr/hal-03585028.