Applying SafeComp, a Formal Integrated System Modeling Framework, to the design of a Steam Generator Controller

Bruno Monsuez
U2IS/ENSTA Paris
Institut Polytechnique de Paris
828 Boulevard des Maréchaux, Palaiseau, F-91120
bruno.monsuez@ensta-paris.fr

Michel Nakhlé
Activité Aéronautique Énergie & Systèmes Industriels
CS Group
22 Avenue Galilée, Le Plessis Robinson, F-92350
michel.nakhle@c-s.fr

Abstract— **We previously introduced in [8] an integrated system of formal model called SafeComp framework that focuses on the implementation of a unified industrial process modeling using the graphic language of Hi-Graphs, a specific class of hypergraphs. This process takes into account that requirements can often be described using different formalisms and additionally provides functional views, taking into account the non-functional and dysfunctional at all stages of the system lifecycle to make the right choices/compromise in terms of software engineering, formal verification and assurance that the system meets the requirements, end-to-end.**

In this paper we show the application of this framework to explore the space of solutions when designing the control-command of the regulation of a steam generator and we also expose the results of this study.

Keywords-Formal System Modeling; Hyper graphs; Multiple-Views Modeling; Solution Exploration; Complex System; Systems Engineering

I. INTRODUCTION

The evolution of complex system design induces an increasing complexity of functionalities that are performed by the system. This increase of complexity impacts the functional, the logical as well as the technical architectures. Additional requirements like safety, dependability, but also reusability or maintainability increase the number of constraints that should be taken into account when exploring the design solution space.

The design of complex systems requires a joint analysis of at least three families of requirements:

- Functional requirements that define the main features of the system;
- Non-functional requirements that define properties such as quality of service and real-time constraints that must be met by the system;
- And Dysfunctional requirements that concern the operational safety, such as reliability, availability, etc.

During the design phase, the exploration of the system space solution must be consistent with the requirements. However, each time a decision on function implementation is taken it may directly impact the system availability. Again a decision regarding system availability may induce new functionalities and will add additional functional requirements. In addition requirements are being described using different formalisms.

For instance, availability or reliability properties are commonly expressed using probabilities. Functional behavior is mainly expressed using transition-based systems (finite state machine, discrete event systems…); Quality of services may be expressed either using a probabilistic model or a set based model. The heterogeneity of the formalism adds additional complexity to the exploration of the solution space.

In [7] & [8], we introduced a Formal Integrated System Modeling Framework as well as it associated methodological process called SafeComp. The goal of this new design methodology is to simplify the design solution space exploration. This methodology not only ensures that the system works according to the expectations but also ensures that the resulting design is optimal, exhibits a safe behavior and is reliable.

In this paper we show the application of this methodology on the design of the regulation control-command of a steam generator. The goal on this study was twofold: we first wanted to see if current control-command design can be easily modeled and explored with the methodology and if the methodology can be easily exploited by system design engineers; we then want to see if doing a joint refinement of functional, non- functional as well as dysfunctional views can help the system designer to explore new non-traditional control-command design implementations that still comply with the safety and reliability constraints.

The paper is structured as follows: after a brief introduction of the SafeComp methodology (Section 2), we briefly introduce the control-command of the steam generator (Section 3). We then present how the SafeComp methodology was declined on this use-case (Section 4) and we present some key results that were obtained using the SafeComp methodology (Section 5). We finally discuss the interest of the approach; we show future works and conclude the paper.

II. INTRODUCING THE SAFECOMP METHODOLOGICAL FRAMEWORK

The SafeComp (Safe & Compositional) methodology proposes a Formal Integrated System Modeling Framework that extends the state charts model introduced by David Harel [6], [5]. Harel's state charts extend the classical state-transition formalism by adding three additional notions: hierarchy, parallelism and diffusion.

Hi-Graphs (as presented in [1], [7] & [8]) extend the state charts by generalizing hierarchy and adding orthogonality. Generalizing hierarchy with respect to the state charts make Hi-Graphs capable to model the refinement process (top-down exploration) as well as the satisfaction process (bottom-up exploration). Multiple abstraction level can be modeled, a super blob provides the information at a given abstraction level, the blobs that are hierarchically connected to the super blob provide the information at a lower abstraction level.

Adding orthogonality allows projecting a given formal object into different dimensions. For instance, the blob expressing a requirement can be decomposed into three orthogonal regions: a first region expressing the physical requirements, a second region expressing the behavioral requirements and a third region expressing the interface requirements. Each region can be seen as the projection of the blob to the dimension expressed by this region.

Since Hi-Graphs are hyper graphs [2] and since hyper graph arrows can connect more than two states, Hi-Graphs allow merging blobs into super blob, limiting the number of states and reducing the impact of state explosion.

A. *Interests of using Hi-Graphs for System Modeling*

Hi-Graphs can capture and express the different representations of a system. With respect to System Engineering, a Hi-Graph can represent either jointly different system regions or separately different system views like, for instance, the structural, the behavioral or the requirements views. Hi-Graphs transformation allows transforming a Hi-Graph that jointly represents multiple paradigms or notions into Hi-Graphs that represent separated paradigms or notions. The inverse transformation is also possible.

Depending on the views, the mapping of the elements to the semantic notions that are captured in the view is different. Table I presents a typical mapping of Hi-Graphs elements to the associated semantic notions that depends on the system view that is represented by the Hi-Graph. This mapping is simply illustrative and depending on the views that are introduced. Additional and different mappings may be dynamically introduced when building and refining the model and there is no need to introduce them when starting the refinement process.

TABLE I. MAPPING HI-GRAPHS ELEMENTS TO SEMANTIC DEFINITIONS

	Requirements	***Structural***	***Behavioral***
Node	• System requirements	• Component attribute • Component functionalities	• System state
Arrow	• Assigning a requirement to a system component • Assigning a requirement to a system behavior	• Heritage • Assigning a component to a system behavior	• State transition
Hierarchy	• Requirements hierarchy	• Allocation of attributes to component • Allocations of functions to component	• Behavior prioritization
Orthogonality	• Logical partition of the Requirements	• Composition • Aggregation	• Concurrent Behavior

Hi-Graphs also support bi-directional refinement. Traditionally, when refining, an element is replaced by a more detailed/specialized element that inherits from the initial element and adds additional information or set of constraints or set of constraints, *the inherited information*. Since Hi-Graphs supports orthogonality and hierarchy, when refining, the element that is refined get replaced by a more detailed element and the additional information get propagated to all the interconnected elements in the different views that are connected to the current element.

Depending on the nature of the information, (this may lead either to an additional requirement (top-down specialization) or to the satisfaction of a property (bottom-up simplification), the element will be modified to take into account the modification and this modification process iterates till no additional changes should be propagated.

B. *The SafeComp Canvas*

The capability of Hi-Graphs to model multiple views allows extending the canvas introduced by K Fogarty [3], [4] with additional constraints when doing System Exploration, among others the non-functional, dysfunctional and certification constraints that the system design should comply with.

The Figure 1 presents the complete System Modeling Canvas as proposed by the SafeComp Framework. The extension to the original Fogarty's original canvas gets highlighted in red and green.

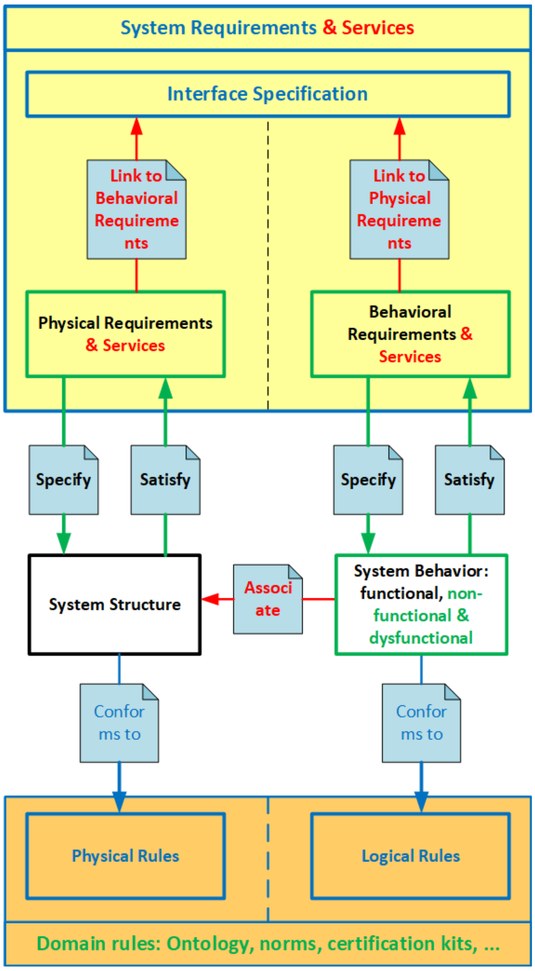
System Requirements & Services
Interface Specification
Link to Behavioral Requirements
Link to Physical Requirements
Physical Requirements & Services
Behavioral Requirements & Services
Specify
Satisfy
Specify
Satisfy
System Structure
Associate
System Behavior: functional, non-functional & dysfunctional
Conforms to
Conforms to
Physical Rules
Logical Rules
Domain rules: Ontology, norms, certification kits, ...

Figure 1. The SafeComp Canvas

III. PRESENTATION OF THE STEAM GENERATOR AUTOMATIC CONTROL REGULATION

Steam generators are heat exchangers used to convert water into steam from the heat produced by the nuclear core. Since steam generators contribute to cooling the nuclear core, steam generators are safety equipment and must be able to operate for a long period after the nuclear reaction has been stopped.

The steam generator automatic control ensures that the steam generator always operates inside the operational limits defined for the nuclear installation. Those operation limits define operational targets on the level of water as well as on the rate of flow.

Typically, a steam generator controller is composed of:

- A continuous measurement system, that implements different sensors to ensure continuous, safe and reliable measurement of the required physical values,
- An electronic computation unit that generates the commands of the remotely controlled valves,
- A set of controlled valves that controls the rate of flow of the water entering the steam generator.

Reliable generation and execution of the regulation commands are required to ensure that the steam generator works in the expected operational domain. Since cooling the nuclear core should be performed during many months after the nuclear reaction has stopped, the expected reliability and availability of this function is very high. Table II summarizes some of the failure modes and the consequences of the failures.

TABLE II. EXTRACT OF THE FAILURE MODES AND EFFECTS ANALYSIS

Errors	*Causes*	*Consequences*	*Severity*	*Prevention & Mitigation*
No flow regulation	• Missing input values (failure when measuring or communicating the values, etc) • Failure when generating the commands (hardware or software failure, etc) • Violation of real-time constraints • No execution of the generated commands (failure of the remote valves, etc)	System may operate outside the valid operational domain	Severe	• Detecting the absence of command generation or command execution and the causes
Erroneous command generation	• Invalid input values (failure when measuring or communicating the values, etc) • Failure when generating the commands (control-command instability, software error, data corruption, etc) • Failure when executing the command (wrong position of the remote valves, etc)	System may operate outside the valid operational domain	Severe	• Monitoring measured values • Monitoring commands hardware & software integrity • Avoiding and detecting software errors • Monitoring the behavior of the valves.

IV. APPLYING SAFECOMP TO THE STEAM GENERATOR CONTROL

We did apply to the Steam Generator Control, the SafeComp methodology step by step as follows:

- **<u>Step</u> <u>1:</u>** We first model the set of requirements of the Stream Generator Control. Those requirements include the safety, reliability and availability requirements as shortly described in the previous section. It also includes the physical requirements regarding flow velocity and maximal and minimal rate of flow. It also defines the constraints regarding the expected precision on measurement as well as on the actuators. It finally defines the time constraints, command cycle & refresh time, execution latency, etc. Figure 2 presents the Hi-Graph that captures those requirements. This Hi-

Graph decomposes itself into two Hi-Graphs; a first one that explicit all the operational constraints and a second that collates all the requirements that the control-command of the steam generator regulator should verify. In those Hi-Graphs, orthogonality maps the requirements to the corresponding Hi-Graph region according to the semantic nature of the requirements.

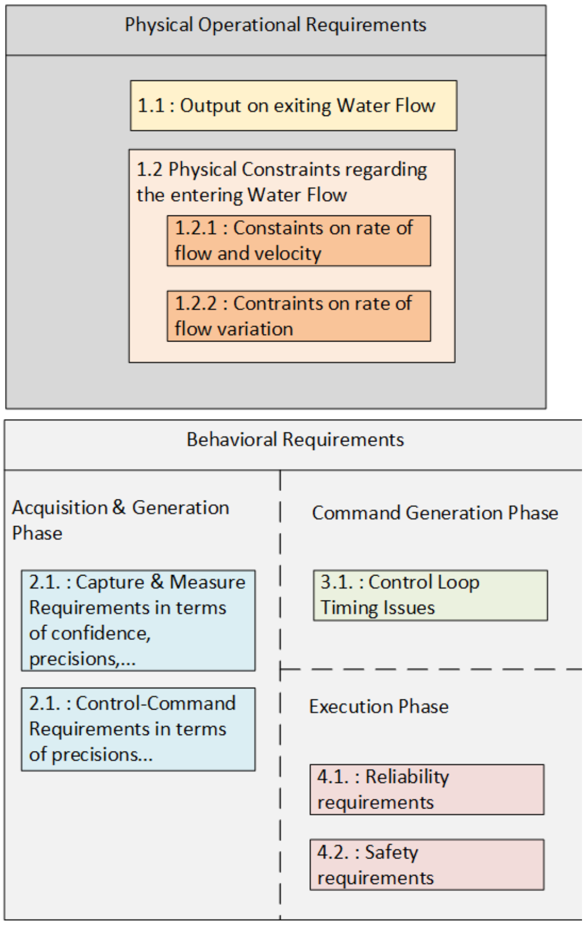
Physical Operational Requirements
1.1 : Output on exiting Water Flow
1.2 Physical Constraints regarding the entering Water Flow
1.2.1 : Constaints on rate of flow and velocity
1.2.2 : Contraints on rate of flow variation
Behavioral Requirements
Acquisition & Generation Phase
2.1. : Capture & Measure Requirements in terms of confidence, precisions,...
2.1. : Control-Command Requirements in terms of precisions...
Command Generation Phase
3.1. : Control Loop Timing Issues
Execution Phase
4.1. : Reliability requirements
4.2. : Safety requirements

Figure 2. Hi-Graph structuring the behavioral requirements

- **<u>Step</u> <u>2:</u>** We proceed with the hazard analysis as presented in the previous section and we create the views associated with the safety and availability requirements. This view is expressed in terms of a Hi-Graph that collates the feared events as presented on Figure 3.

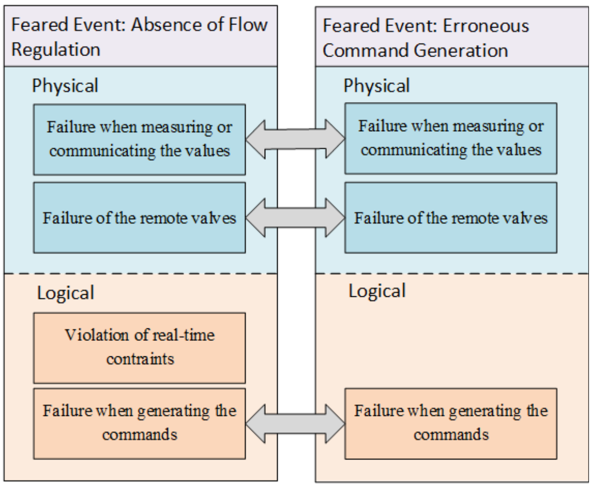
Feared Event: Absence of Flow Regulation
Physical
Failure when measuring or communicating the values
Failure of the remote valves
Logical
Violation of real-time contraints
Failure when generating the commands
Feared Event: Erroneous Command Generation
Physical
Failure when measuring or communicating the values
Failure of the remote valves
Logical
Failure when generating the commands

Figure 3. Hi-Graph capturing the feared events

Each initiating event that may lead to the feared events is mapped into a region that collates all the causes of the same nature. On Figure 3, two regions have been illustrated; a first region that corresponds to the physical failures, failures due to a sensor or an actuator and a second region which collates the logical failure due to violation of the real-time constraints, potential computational errors and so on. We also see that a sub Blob may be shared between two super Blobs.

<u>Step</u> <u>3:</u> We then introduce the structural and the functional views modeling the functions that the steam generator controls. Orthogonality plays an important role since it allows decomposing the model into orthogonal sub-models. For instance, we introduce three orthogonal regions that correspond to the different operational phases of the control-command loop, acquisition, command generation & command execution. Each sub-region will also be decomposed into a "processing region" and a "communication region" as represented in Figure 4. Since each sub-region contains a "communication" component, mapping the Hi-Graph to the "communication region" extracts all the "communication flow" between all the functions that composes the steam generator controls.

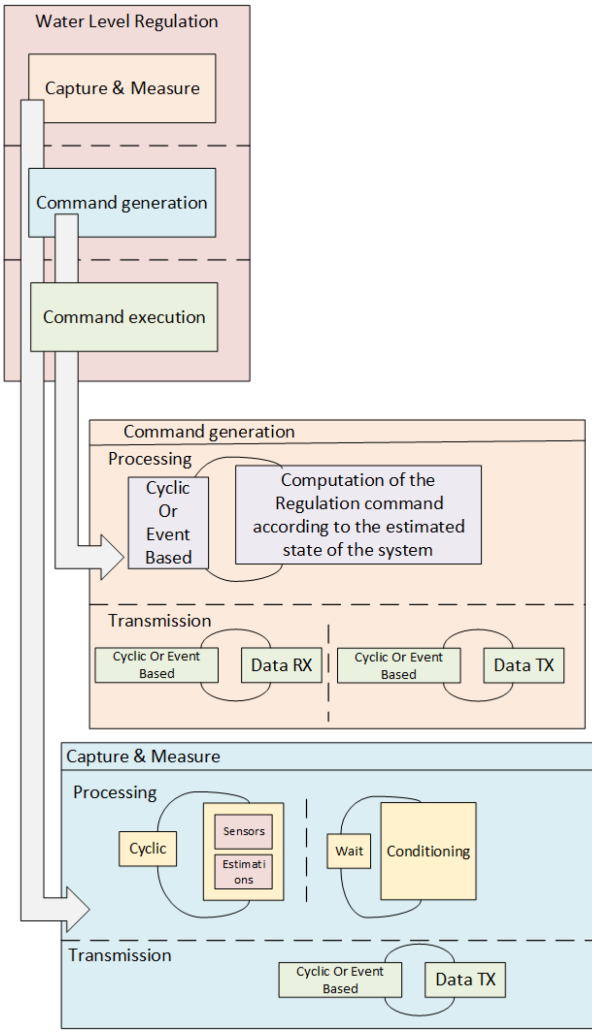

Figure 4. Hi-Graph defining the functional & structural view of the steam generator control

- **Step 4:** We then create the relations between the functional views and the non-functional and dysfunctional views as created during Step 1, and Step 2 and Step 3. For instance, the feared event that consists in generating an erroneous command will be refined according to the structural and functional views regarding the acquisition phase. This will lead to the Hi-Graph presented in Figure 4. Again orthogonality plays a key role since this allows separating the cause of the dysfunction between logical (software error, data integrity failure), physical (hardware error, sensor drift) and temporal errors (delayed data communication, violation of real-time boundaries, etc.). In addition to the refinement of the Hi-Graphs that represents the feared events; we introduce the Hi-Graphs that map the reliability requirements as well as the loop requirements to the different components of the system as presented in Figure 5.

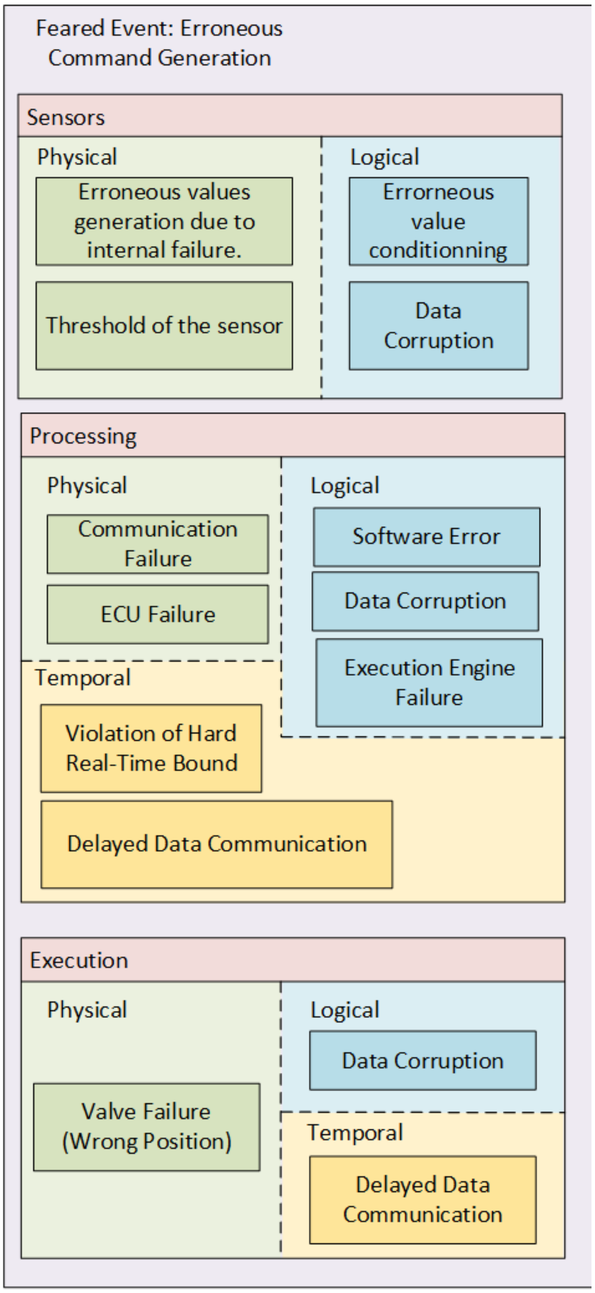
Feared Event: Erroneous Command Generation
Sensors
Physical
Erroneous values generation due to internal failure.
Threshold of the sensor
Logical
Errorneous value conditionning
Data Corruption
Processing
Physical
Communication Failure
ECU Failure
Logical
Software Error
Data Corruption
Execution Engine Failure
Temporal
Violation of Hard Real-Time Bound
Delayed Data Communication
Execution
Physical
Valve Failure (Wrong Position)
Logical
Data Corruption
Temporal
Delayed Data Communication

Figure 5. Refining the Hi-Graph representing the feared events according to the functional and structural views

- **Step 5:** We start the space solution exploration. We refine and propagate the refinement between the different views, taking some decision about implementation.

For instance, in Figure 6, we proceed with the refinement of the acquisition of the water flow that enters into the steam generator. We must take into account (1) the expected precision as defined by the requirements, (2) the safety & reliability levels according to the normative requirements. The acquisition decomposes itself in two phases: (i) the acquisition of the value from the sensors (ii) and the conditioning of the value to be sent to the processing unit. Since the required safety and reliability constraints imposes to estimate the precision of the sensors state as presented in Figure 7, to monitor the state of the sensors and to at least triplicate the sensors leading to the refined GRAFCET presented in Figure 8.

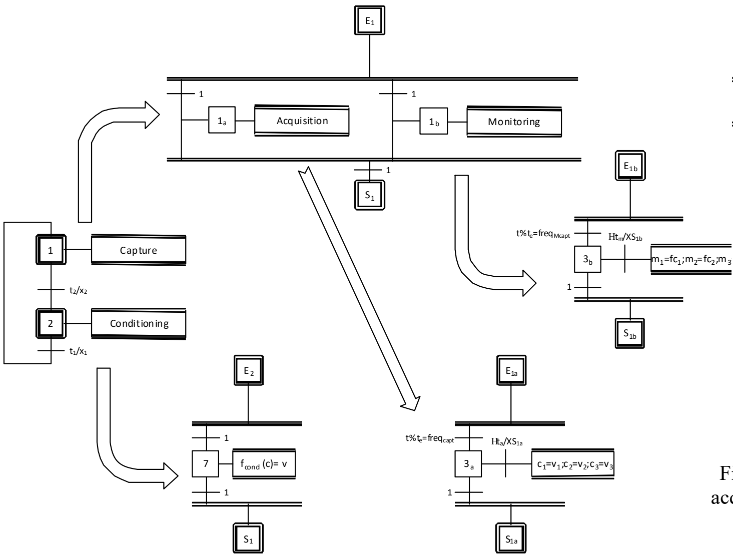

Figure 6. Refinement of the GRAFCET that represents the measure of the water flow and the conditioning of the measure

.

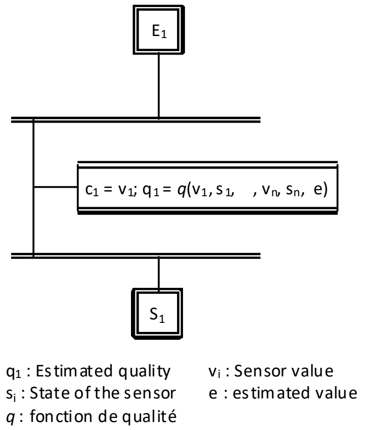

Figure 7. Computation of the quality of a value

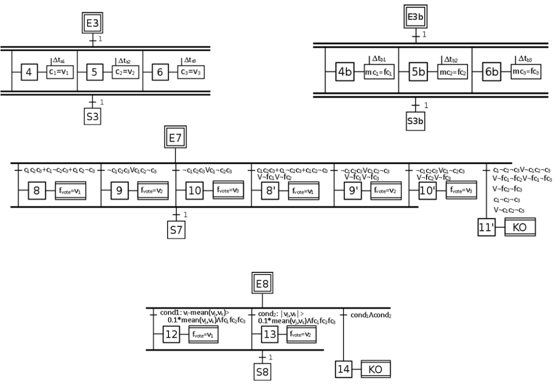

Figure 8. Refinement of the GRAFCET that introduces the multiple acquisitions and the voting logic to comply with the safety & reliability requirements

The next step consists in mapping the different value conditioning and monitoring functions to the type hardware that will implement them. In Figure 9 we map those functions to the components on which the functions will be instantiated. The measure acquisition stage get mapped to ‘’Physical Sensors’’, the monitoring and the voting stages get mapped to ECU modules.

When refining, we need to introduce additional views regarding the different services that should be offered (transmission service, computation service, voting service, monitoring service) by the hardware modules that host the functions as well as the requirements that are associated to those hardware modules in terms of safety and reliability (see Figure 10)

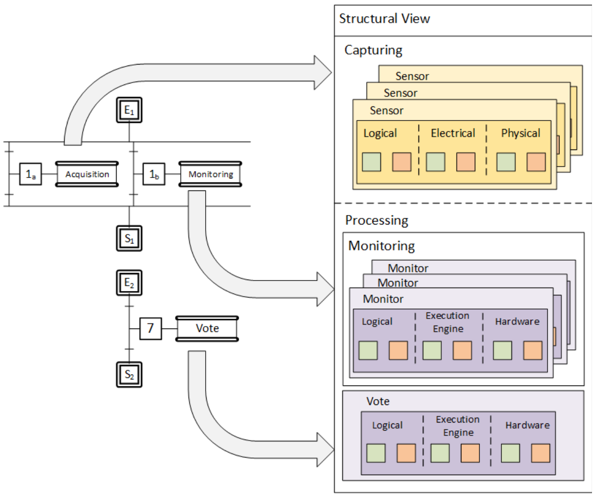

Figure 9. Mapping the functions to hardware structural views

- **Step 6:** We consolidate the architecture and synthesize the functional and dysfunctional properties. During this phase, we finally map the functions and services to the different computation node with respect to the functional, non-functional and dysfunctional constraints as well as with respect to hardware capabilities and system ontology's.

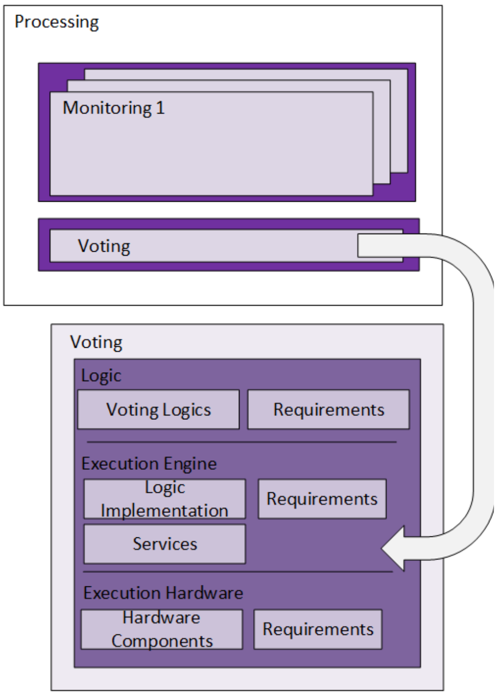

Figure 10. Mapping to implementation & requirements views

V. RESULTS

The application of SafeComp methodology to the design of the steam generator control system allowed us to explore the space of the solution taking into account the functional requirements, the specific constraints on the reliability and maintainability of the systems (the sensors must be replaced during a periodic shutdown), hardware performance (sensors are sensitive to drift, available computing power), the complexity of the automatic control logic, the reaction time of the system.

Exploration of the solution space resulted in a different set of implementations, all of which provided the required level of security or reliability.

All solutions require at least two voting steps to ensure faulty sensor detection and safe command-and-control generation. This is in line with state-of-the-art control-command architectures that have been deployed or deployed in the Generation III nuclear reactor.

However, the use of the SafeComp methodology over current system engineering approaches, which introduce strict separations between different views, has many advantages. Most importantly, every time a modification or refinement decision was made, the consequences of this decision were officially propagated to the blobs that capture the other aspects of the design.

As a general rule, when designing the monitor "monitoring" the sensors that measure the water flow, it is recommended to have an estimator providing the expected flow according to the order; this has an impact on the control logic, the design of its implementation, its safety and reliability...

The second advantage is the ability to dynamically introduce additional orthogonal dimensions to structure the exploration, if necessary. For example, when software implementations of functions are refined, we may decide not to directly map software implementations to the compute nodes that will perform the computation, but to map the software to a virtual runtime region that will then be mapped to the nodes of calculation. This approach makes it possible to explore new solutions that would have been more difficult to explore in the case of set of predefined views.

Finally, one of the good results we have achieved is that instead of splitting the different voting services onto additional hardware as is currently done, voting services can be implemented on the electronic control units that perform the calculation (see Figure 11), allowing a hardware architecture much simpler than the one currently used and offering the same level of security and reliability.

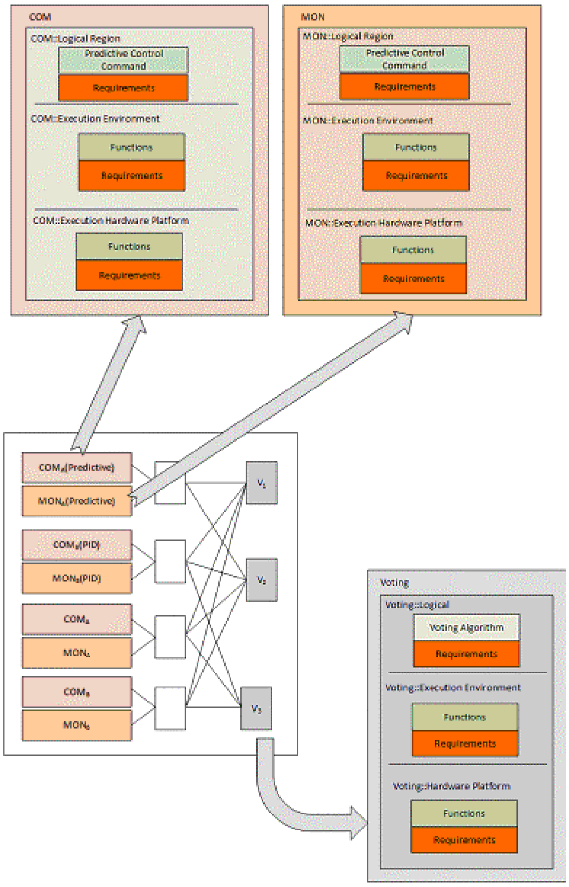

Figure 11. Final multiplexed control-command controller architecture

VI. CONCLUSION

Within the DEPARTS project, studied use cases were: 1) A Vehicle Electronics Loading Ramp; 2) A 3-stage airborne micro-launcher (new generation modular architecture space system); 3) One-lane and two-lane controlled Railway track Cross Section (RCS); 4) Railway Switching Station; 5) Regulation control-command of the steam-generator.

In [8] we considered a controlled Railway track Cross Section (RCS) throughout the paper to illustrate how to use Hi-Graphs to model and refine a typical system.

The design of the regulation control-command of the steam-generator did demonstrate the interest of performing a joint exploration of the solution space with respect to functional, non-functional and dysfunctional properties. The complexity of the exploration was not higher than separating the different concerns and analysis them separately. In addition to the current architectures, new original and smart solutions did emerge when performing the analysis.

The lessons we have learned are that formal techniques can indeed be applied successfully in industry and can be both efficient and effective. Thanks to SafeComp, these techniques allow reaching high level of quality for safety critical systems. For critical systems, while the global cost is the same than usual techniques, such as thorough testing, the quality of the resulting artefact, including its documentation is higher.

Future work will go in two directions: first to develop a tool chain to manipulate and transforms the Hi-Graphs; secondly to see how easy adding new paradigms like, for instance, cyber-security requirements and properties can be achieved.

ACKNOWLEDGMENT

This research carried out in the Project DEPARTS was partially supported by grants from BPI France within the French government R&D program PIA/FSN.

REFERENCES

[1] H. Aboutaleb, "*Applying Hi-Graph-Based Model to System Engineering - Methodology, Formalism and Metrics*". PhD in Computer Science, Paris, Ecole Polytechnique, 2015.

[2] C. Berge, "*Graphes et Hypergraphes*", Dunod, Collection Monographies Universitaires de Mathmatiques n°37, janvier 1970.

[3] K. Fogarty, "*System Modeling and Traceability Applications of the Hi-Graph Formalism*". MS thesis. Institute for Systems Research. University of Maryland, MD 20742. May 2006.

[4] K. Fogarty, M. Austin: "*Systems Modeling and Traceability Applications of the Higraph Formalism*", In Systems Engineering: The Journal of the International Council on Systems Engineering, Vol. 12, No. 2, summer 2009, pp. 117-140

[5] O. Grossman and D. Harel, "*On the Algorithmic of Hi-Graphs*". Technical Report CS97-15. The Weizmann Institute of Science, Department of Applied Mathematics and Computer Science. September 1997.

[6] D. Harel, "*Statecharts: A Visual Formalism For Complex Systems*". Science of Computer Programming 8 (1987) 231-274. North-Holland.

[7] M. Nakhlé and B. Monsuez "*RTCE-SafeComp: Introduction à la Méthodologie SafeComp et Rapport d'étape*", Version 1.3c du 11/9/2018.

[8] A. Otmane Cherif, B. Monsuez, M. Nakhlé, V-A. Paun, "*Using Hi-Graph to define a Formal Integrated System Modeling Framework that ensures Complete System Consistency*", in 26th International Conference on Systems Engineering (ICSEng), December 2018, Sydney, Australia