Dependability analysis activities merged with system engineering, a real case study feedback

Robin Cressent, Pierre David, Vincent Idasiak, Frédéric Kratz

Dependability analysis activities merged with system engineering, a real case study feedback

R. CRESSENT
PRISME / ENSI de Bourges, Bourges, France

P. DAVID
Grenoble-INP / CNRS G-SCOP UMR5272, Grenoble, France

V. IDASIAK & F. KRATZ
PRISME / ENSI de Bourges, Bourges, France

ABSTRACT: This article largely illustrates the deployment of safety and dependability analysis of a critical aircraft system within a Model-Based System Engineering context. The level of analysis provided here is two-fold. A first part of the study examines the process level integration between the traditional System Engineering activities and the safety and reliability assessment operations. Models of engineering activities are given and merged to produce a guideline for the project development and highlight the knowledge management. The second level of this study exemplifies the application of the preceding principles on the product development through the feedback and experience learnt on a ramjet powered vehicle embedded system. This extended instance shows the diverse knowledge materials created throughout the project as well as their imbrications and logical chaining. We mention also how the produced results help in piloting the advances and the validation needs (especially for real time constraints evaluation).

1 INTRODUCTION

The current highly competitive context imposes even more rapid and low cost development phases, for the creation of even more efficient systems. Therefore, the designers' challenge is to master complexity in a restrained time, while validating performances. The performances validation process is very important for safety-relevant applications or critical systems. Thus, elaborated analysis techniques have to be used to assess those requirements. These are dealing with state reachability proving, dysfunctional behavior simulation and indicators quantification. Moreover it has been proven that it was cost-effective to detect potential failures as soon as possible, since it permits to avoid reengineering costs.

The System Engineering (SE) process deployed for the design of such systems have to organize all the tasks to be performed from the definition of the functional architecture to implement, to the validation of the dependability and safety. Currently, numerous standards and specific engineering methodology have been proposed to tackle this issue (e.g. IEC 61508 and its derivative). Nevertheless the difficulty that remains for developers is to optimize the integration of safety and reliability analysis in those processes to minimize the time consumed by those activities.

To lessen the time needed to conduct jointly the SE and Dependability processes, it is necessary to identify and model the activities that compose them to plan their interactions and to pilot the whole. In this article, we adopt the definition of "process" described by the INCOSE Handbook (INCOSE 2004): "a process is a set of interrelated or interacting activities which transforms inputs into outputs". A process doesn't define how activities transform the inputs but what is exchanged along those activities.

Efficiently conducting the SE and dependability processes is a topical subject in current research projects. In fact, the introduction of new standards for critical and safety relevant systems imposes on systems manufacturers to master a new kind of Engineering process including safety management aspects. The aim of projects as CESAR (David & Shawky 2010) and SASHA (Langheim et al. 2010) is to find an efficient manner to follow the safety lifecycle described in the standards and to furnish the adapted tool supports and the tool interactions to implement them. In those projects, specific processes are studied and we can note that most of engineering domains (nay each manufacturer) are using their specific processes, lifecycles and dependability analysis baselines. Therefore we propose in this article a generic approach to help in merging

and supporting SE and dependability processes. This work will be illustrated by a two levels example extracted from the case study of the LEA project. A first part (section 2), illustrating processes management at a project level showing the fusion of both SE and Dependability studies activities. And the second part (section 3), showing the product level application of the defined processes. This example will also underline the central role of repositories making the junction between the interleaved activities at the project level and the coherence and traceability of the produced data at a product level.

One of our current project is the LEA project started in 2003 by MBDA-France and ONERA to address the key issue of the aeropropulsive balance of a dual-mode ramjet powered vehicle in the range Mach 4 to 8 (Falempin & Serre 2009). A development methodology has been defined for such type of vehicle, together with the numerical and experimental tools enhancement to enable predicting the flight performances with suitable accuracy. This methodology is now being applied to minimal size experimental vehicle, called LEA, which has passed the preliminary design review in 2006, and the critical design review in 2009. Finally, several flight tests will be performed at the end of the program to validate the quality of performance prediction. Four flights are planned, and will be performed between 2013 and 2014, operated from Russian test range and using Russian hardware for initial acceleration. Our team is charged of specifying, designing, testing and validating the embedded system, which must control the flight from the launching of the craft to the final crash. Moreover, the embedded system must control the dual-mode ramjet carburetion process and some safety functionalities, like the auto-testing function for automating GO/NoGO decision before launch or the detection of separation from booster and the ignition order. The LEA project gives us an experimental platform to adapt our work to embedded system.

2 PROJECT LEVEL

At the project manager's level, merging SE and Dependability processes begins with the identification of each activity performed through those processes. This leads the project manager to plan SE and dependability activities together showing the need for tools to fill the gap between them.

2.1 *Modeling SE and Dependability processes*

In order to underline the crucial points of processes merging, we model the activities which compose them by using UML activity diagrams. We provide a generic model of purely SE so that each of our industrial partners can duplicate this model into their own industrial SE processes. We thus specified the SE process provided on Figure 1. In this UML model, SE activities are modeled as *activities* and the data generated are modeled using *Data Store Nodes*. In this article, *Data Store Nodes* represent a certain view or information available in a model. They match the UML 2 (OMG 2009) definition: “A data store node is a central buffer node for non-transient information”.

Figure 1 represents the model of specification and design SE activities:

- Define the system’s objectives.
- Establish the functions of the system.
- Specify the required performances (requirements and constraints).
- Determine the physical and logical architecture.

Those activities will product knowledge easily modeled by using SysML. As said before, the pieces of knowledge are represented by *Data Store Nodes*. The generic version of this model didn’t implemented flow control nor feedback and transfer of data between activities as those are tied to the industrial background.

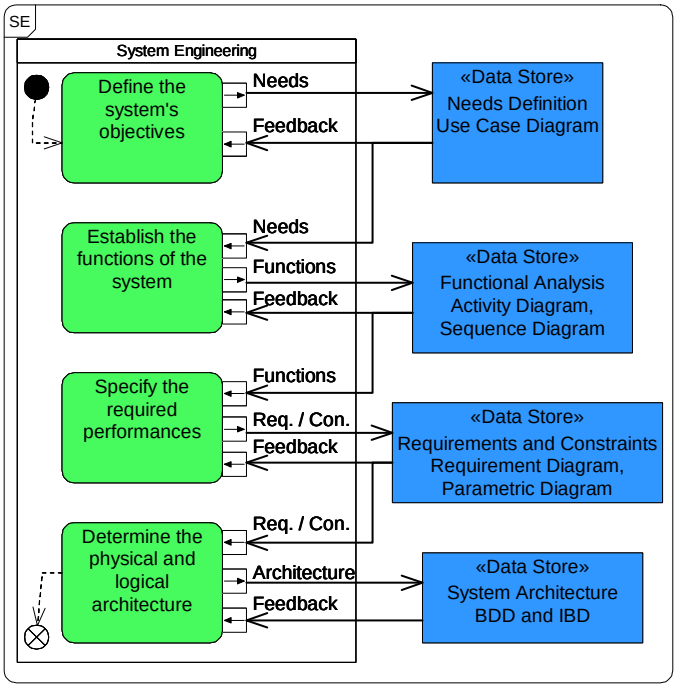

Figure 1: System Engineering process

Each activity can be iteratively conducted until a satisfying level of detail or quality, have been reached. For each iteration, former data contained in data stores nodes is replaced by updated data.

We then modeled the dependability activities in their process and determined its needs in terms of knowledge management. We present on Figure 2 an example of dependability analysis subprocess: performing an FMEA. The *data stores nodes* in this di-

agram represent the pieces of information needed to initiate those activities. We can observe links between the needed knowledge in this FMEA process and the knowledge produced by the SE process. The *data stores nodes* "Functional Analysis" and "Architecture Design Choices" are for example linked to the *data store nodes* "Functional Analysis" and "System Architecture" described in the SE process model. We can also observe that some data store nodes such as "Feedback Database" and "Expert's Knowledge" can't be linked to pieces of information produced by the SE process. It reveals one of the issues that our methodology tries to address: the dispersion of valuable knowledge in the global process.

The "Functional Analysis", needed in the FMEA process is an output of the SE process, stored in sequence diagrams and activity diagrams. In the same way, the "Architecture Design Choices" can be presented in block definition diagrams and internal block diagrams produced during the SE process. Furthermore, requirements and constraints will be useful to the expert to evaluate the gravity of the failure modes described in the FMEA. Those observations forms the first issue addressed by our methodology: Connecting the SE and dependability processes and supporting an automatic transfer of the shared information.

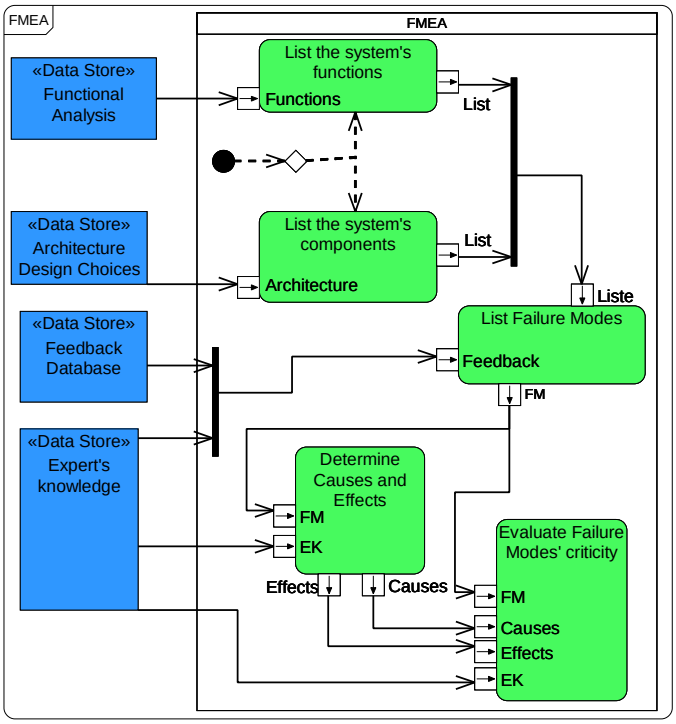

Figure 2: FMEA process

2.2 *Our solution : the MéDISIS Framework*

Our methodology called MéDISIS (Fig. 3) is centered on a Dysfunctional Behavior Database (DBD) which stores dependability feedback and offers a set of translation processes from a main system model in SysML to different specific modeling languages to perform dependability studies. (David et al. 2010) (Cressent et al.2011).

MéDISIS offers tools to help managing data and knowledge through the whole project, including their creation, expression, analysis, perenniality and re-use. Those tools and resources form a coherent framework and aim at reaching the following objectives:

1- Easy the knowledge transfer between teams and among the various engineering levels.

2- Speed up the dependability analysis.

3- Organize the common use of information using models.

4- Permit the re-use of knowledge between projects (i.e. easy the use of feedback information).

5- Identify the needs, plan, and store the results of the project's activities through all the life cycle.

6- Increase the coherence and the quality of the dependability analysis.

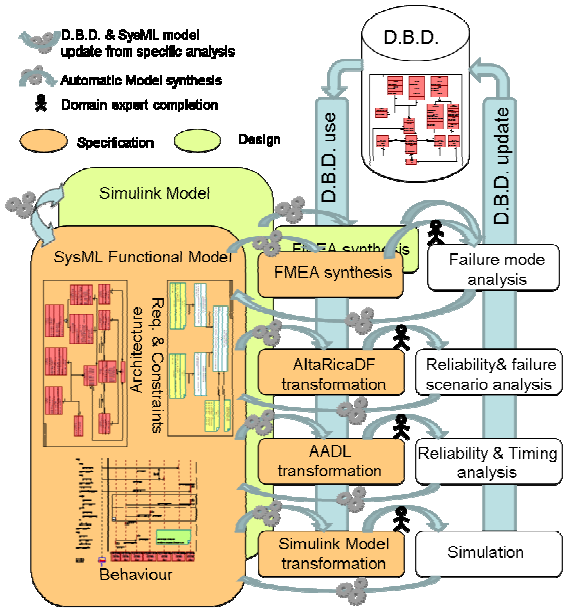

Figure 3: The MéDISIS framework and its processes

Natively, using SysML as the central model language benefits the objectives 1 and 3. Indeed, SysML permits multi-view modeling which address the expectations of every actors of the system design. The other benefit brought by SysML is the possibility to model requirements which create a support to their traceability through the model and project evolutions and address the objective 5.

As described before, MéDISIS is centered on a database destined to assure the perenniality of the information. This DBD contributes to address the 1rst objective. It permits to manage knowledge, brought by each specific expert during dependability analysis, in a single organized structure. The DBD metamodel lies on SysML meta-classes and is partially translated into other language through certain

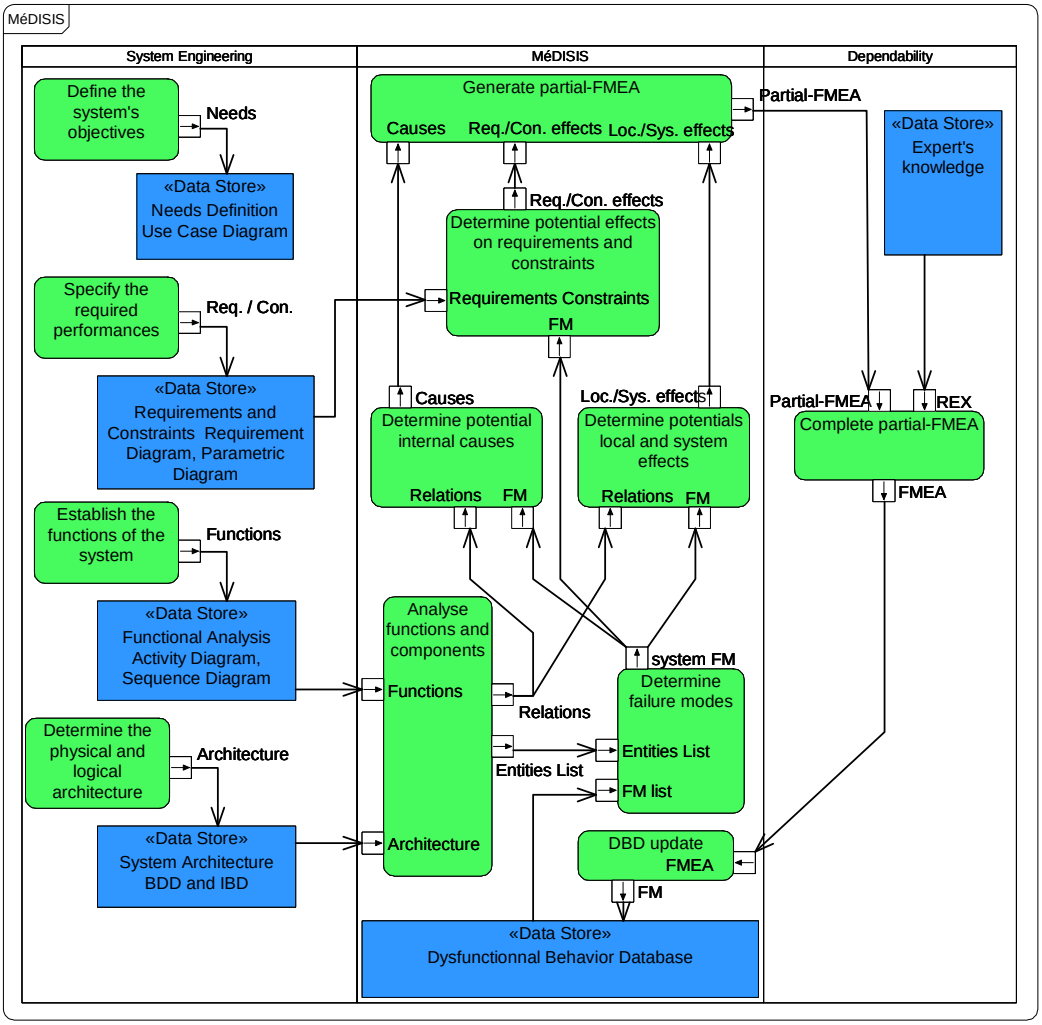

Figure 4: MéDISIS activities for FMEA generation

MéDISIS processes. The multi-view and multi-language aspects of this database address the objectives 2 and 4 since it permits to access quickly dependability information, to be used or to be stored. Finally, the DBD enhances the coherence between dependability analyses thanks to its structured meta-model which link together functional information, domain specific knowledge, and dependability results, addressing by the way the objective

Beside the DBD, MéDISIS is composed by a set of translation processes offering a way to transform a SysML model into other ones using specific languages. Those processes are automatable, proof of their coherence, their fast execution and of the traceability of the data treated, created and re-used. Those processes address the objectives 1, 2 and 6. Every one of them aims at generating partial models in the target language, to be completed by dependability experts. They fulfill the need of our industrial partners that use various tools and formalisms along their projects, manipulated by different experts in their own domain.

Currently, MéDISIS counts 4 processes that translate SysML models into target languages. The benefits that bring SysML are described in (Cressent et al. 2010). The creation of the DBD and its meta-model, the FMEA generation process and the Altarica DF translation process were described in (David et al. 2010).and (Cressent et al. 2010), described the SysML to AADL translation process and the process to Simulink.

We can observe in Figure 4 the result of the FMEA generation process at a project level. It builds the desired bridge between SE and dependability activities and exemplifies the merging process we were able to perform by analyzing the separated activity diagrams (Fig. 1 & 2). Each MéDISIS process can be modeled in the same way. Ultimately, it furnishes to the project manager a method to plan the design process taking into account both the SE and the dependability aspects of the project. This plan is then followed creating a workflow at the product level that will benefit from the MéDISIS tools.

3 PRODUCT LEVEL FEEDBACK

In this section, we present our feedback on the LEA project supported by the MéDISIS framework, at the product level. The first process we used were the FMEA generation, during an early dependability and

safety analysis. Then we employed the AADL process to help choosing the best architecture of the system. The FMEA process is then re-applied to detail the study, taking into account the architecture and the components, to precise the dependability analysis. Before the design phase, we translate the system model in Simulink (Cressent et al. 2011). The resulting model allows us to simulate the system to get information about error propagation early in the design process by performing fault injection. We focus our thoughts on an internal communication function between the flight controller and the acquisition module.

3.1 *LEA overview and feedback*

During the flight, LEA will be carried by a plane, dropped and then propelled by a booster before being able to fly autonomously. Numerous functions must be guaranteed, such as the auto-test, detection of the drop, the gas regulation, detection of the end of mission and emission of the data. The benefits of bringing SE and reliability closer were already witnessed during the early design phases of the product. The workflow followed by our team from the specification to the preliminary design is summered and the benefits are highlighted in the rest of the article.

3.1.1 *Getting valuable knowledge from the technical specification of the project*

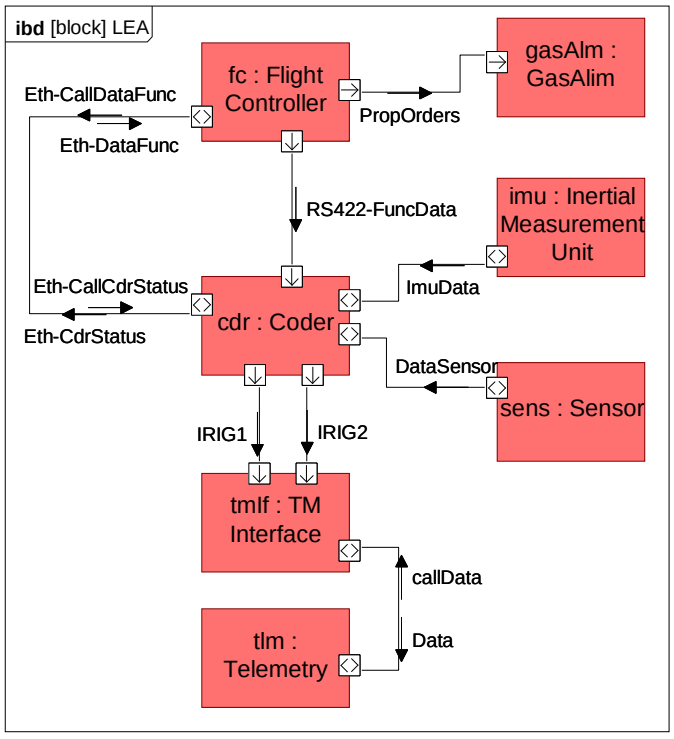

Figure 5: Internal bloc diagram of the LEA vehicle

This step is one of the classic step of any SE process (Figure 4). It is supported by the SysML tool: Requirements formulation (Requirement Diagram produced by the activity “Specify the required performances”), Needs classification (Use Case Diagram produced by the activity “Define the system’s objectives”), synthesis of the technical specification per each use case (Sequence Diagram produced by the activity “Establish the function of the system”). The environmental and technological constraints are qualified using parametric diagrams produced by the activity “Specify the required performances”. Finally, the organic description of the system is realized (BDD and IBD) by the activity “Determine the physical and logical architecture” and then the allocation of the requirements is performed in each view of the system. For example, Figure 5 details the architecture of the LEA vehicle. It presents its major components (e.g. Flight Controller, Coder, Inertial Measurement Unit, …) and their connections. This example highlights the flows exchanged by the components by using SysML *item flows* (e.g. CallData, ImuData, PropOrders, …).

3.1.2 *System analysis.*

The benefit of using SysML during this stage is the ability to inter-connect the analysis and the different entities described in each view through parametric diagrams. The information and knowledge of our system are synchronized bringing more coherence in the model.

3.1.3 *Risk analysis.*

This stage starts with the generation of the partial-FMEA (Fig. 6). The excerpt of FMEA presented in Figure 6 utilizes the notation of the SysML model and has specific writing rules peculiar to the MéDISIS generation process. For example, requirements names are given in brackets in the “Requirement effects” column. The “Causes” column shows the *flows* incoming to the component, the *flow port specifications* that describe this flow and eventually the *constraints blocks* attributed to theses parameters. Since our DBD concerning the technological parts of the vehicle is new, generic failure modes are generally used. Along the progresses of the analysis, the results are introduced in the DBD, increasing the number of failures modes recorded within. Then, at each evolution of the system model, a new partial-FMEA is generated. Finally, a list of the critical threats is created, identifying the severe risks and the components, functions and requirements threatened. The number and the nature of the requirements threatened counts in evaluating the criticality of a failure mode.

In our latest project, during this phase, some subsystems, such as the inertial measurement unit, were identified as critical. The placement of the inertial measurement unit in the functional chain of the system plays a great role in the dependability of the mission. By following some rules and using SysML artifacts (parametric diagram, rationales,…), it is possible to gather and classify the parameters that

Name	Failure Mode	Causes	Local Effects	Requirements Effects	System Effects
Flight controller	Scheduling Failure	Ethernet flow[Env. Constraint : vibration] > [Port Specification]	Propulsion orders [GasAlim] / Functionnal Outputs[Coder] / ConstraintBlock[Data Age]	[Timing requirements] Real-time constraints not fulfilled	Loss of data / Risk of engine failure
		Internal Overstress	Propulsion orders [GasAlim] / Functionnal Outputs[Coder] / ConstraintBlock[Data Age]	[Timing requirements] Real-time constraints not fulfilled	Loss of data

Figure 6 : Excerpt of LEA FMEA

influence reliability, such as the product's life cycle, the mission profile, the use conditions or overstresses. All this information can be stored in the system model and in the DBD. The main advantage of the DBD update is that the collected information becomes available to automatically refine the FMEA. Furthermore, since the DBD is modeled with SysML, we can apply on it other MéDISIS translation processes. This enables us to use the FMEA results into other languages, for instance it permits to realize failure mode injection in a Simulink model.

3.1.4 *System analysis*

The integration of the risk analysis results conducts to reinforce and modify the requirements and constraints applied to the system. These are related to the identified failure modes. This results in modifying the requirement diagrams of our system, and modeling new constraints using parametric diagrams.

By taking into account the new requirements, constraints and the rationales applied to some subsystems, various slightly different architectures are conceivable. The criterions to be analyzed are functional, economic, safety related and temporal. At this stage, we deal with the functional criterions (Cressent et al. 2011).

The estimation of these criterions may cause new analyses needs. For the functional placement of the inertial measurement unit, it appears that the processing time is important. SysML does not permit to model efficiently those aspects. So, we need to resort to a more detailed formalism to deal with the modeling of temporal, architectural and dependability aspects. The example is continued in the following section.

3.1.5 *Specific technical analysis.*

As described previously, certain results brought by the FMEA need to be further analyzed thanks to more specialized formalisms, for instance in the domain of timing constraints. The MéDISIS framework contains a process to generate an AADL model (Cressent et al. 2010). AADL combined with the scheduling tool: Cheddar (Singhoff 2007) permits to find specific temporal constraints stored back in the system model using parametric diagram (Fig. 7). The parametric diagram in Figure 7 presents the relations between components time parameters (e.g. Tad, T_sampl, T_DR_co,…). Theses parameters are bond through *constraints properties* that defines the physics of their relations (e.g. Formula : Control Data Age). Then, we apply this translation process to each conceivable architecture. It permits to compare the temporal constraints of each architecture and to have a criterion to compare them and decide which one is the most appropriate for the needs of the mission. The main objective of this step was to qualify the temporal constraints (Fig. 7). In fact, the quantification won't be possible until the detailed design stages of the product.

In fact, the placement of the subsystems we described earlier impacts the age of the functional data used to regulate the gas in the engine. For some architecture, physical or logical redundancies bring multiplicative factors on processing times. We had to estimate and judge whether performance or dependability must be favored. When the choice is done, the system model is updated, the requirements are detailed with timing information and the new timing constraints are modeled with parametric diagrams. And this knowledge generated by the AADL study permits to refine some failure mode effect, such as a partial loss of connection between subsystems.

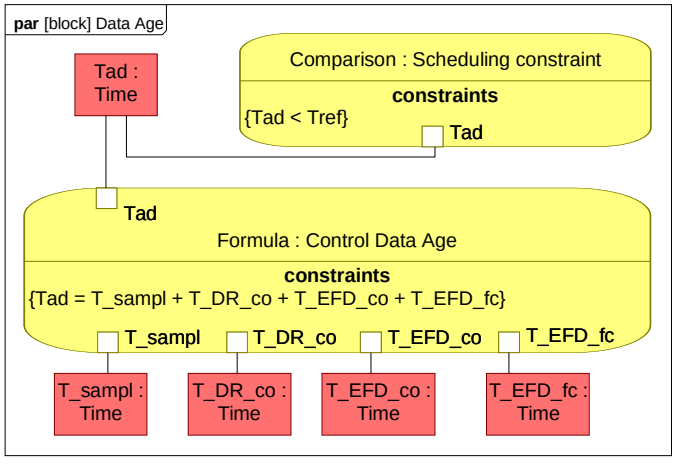

Figure 7: Parametric diagram describing the formula of the age of the functional data.

3.1.6 *Failure mode injection and test cases*

After another phase of FMEA updating, which consists in taking into account the new qualification of the effects studied with the AADL model. The level of detail (i.e. of the system's functioning) allows us to classify again the failure modes with respect to the risk they represent for the system mission. For

the most critical ones, it is necessary to deepen the study. We want to quantify the constraints modeled with parametric. Thereby, the management of the tests is supported by the parametric diagrams. We use fault injection in a Simulink model to understand the dynamic behavior of our system in case of failure.

Indeed, the similarities between SysML and Simulink described in (Cressent et al. 2010) and (Snyder et al. 2010) permitted to define the MéDISIS bridge to the detailed design phases of the product. At this stage, we inject failure modes into our model to simulate its behavior and validate the design choices. Furthermore, tests cases defined using sequence diagrams (Fig. 8) and parametric diagrams (Fig. 7) are introduced as well. Since sequence diagrams permits to define the exchanges of messages between components in the functional system model of the LEA vehicle, it also allows describing non functional exchanges such as test case exchanges. Sequence diagrams will help defining the test procedure when the parametric diagrams will help defining the success criteria. Figure 9 shows an example of Simulink block success criterion implementing the formula presented in figure 7 that is used to validate the system design model. The Simulink design model of the system is then simulated to give us precious information about his behavior in failure conditions. It is then updated taking into account the performances of the system and the effects of failure modes.

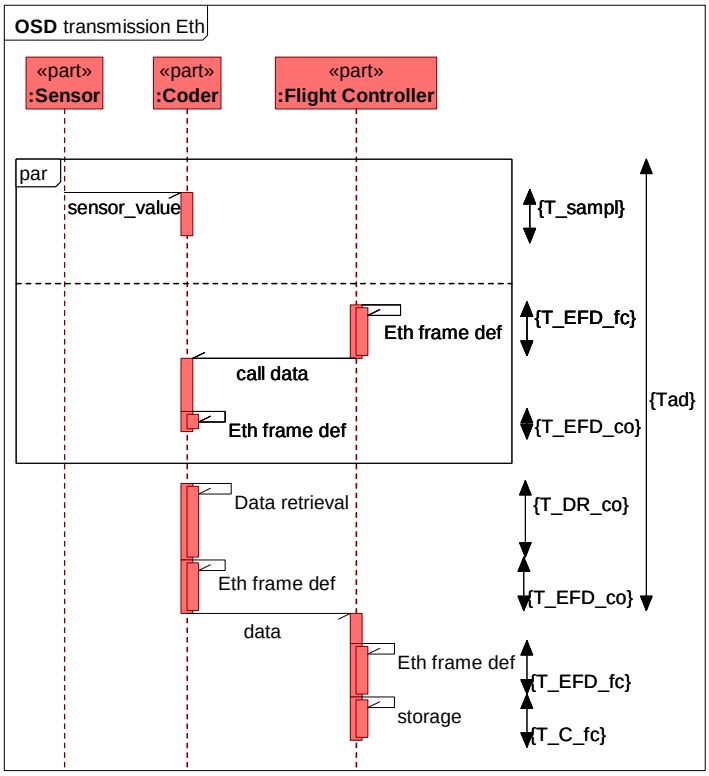

Figure 8: Sequence diagram describing the Ethernet connection between the flight controller and the data acquisition unit.

For our example, we injected a failure mode jamming the data exchanges between subsystems. The parameters of bloc simulating the failure were determined by the FMEA which linked this failure mode to temperature and vibrations overstresses. Results of the simulation were compared to the parametric diagrams modeling the temporal constraints of the communication between those subsystems which can jeopardize the emission of flight data to the ground. Since the main objective of the LEA project is collecting flight data, those failure mode were considered critical. The final results of this fault injection phase were described in detail in (Cressent et al. 2011).

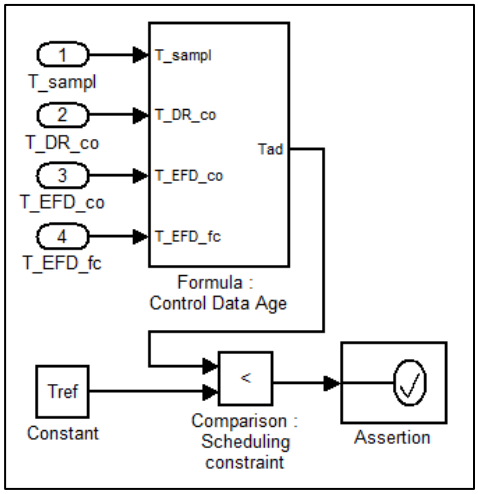

Figure 9: Simulink block defined from the parametric diagram (Fig. 7) used as success criterion for the LEA design model

4 CONCLUSION AND PROSPECTS

Since nowadays systems are complex and multi-technologic, it is tedious to master their design and optimization within a clear and effective Engineering process. We consider that adopting a model based approach is the best way to handle the complexity issues, the exchange of model information and parameters and the communication amongst system level description (e.g. from domain-specific tools to a system level view). We are building the MéDISIS framework in order to help systems engineers deploying a model-based approach for safety critical complex systems.

Currently, MéDISIS is applied in the LEA project, from the project activities planning to the product design phase. Through all the steps performed until now, MéDISIS brought fluency between activities and opened new ways of integrating dependability studies to the system engineering process. At the project level, the possibilities offered by the MéDISIS framework allow to plan SE activities and dependability activities at the same level. Indeed, MéDISIS offers bridges to cross the gap that usually separates them. At the product level, through specification and design phases, the MéDISIS

processes enable and ease the connection between tools (e.g. SysML, FMEA, Simulink, …). Furthermore, MéDISIS increases the consistency of the system model and dependability analyses by synchronizing the information from any language into the system model and the DBD.

One of the remaining issues that we need to address is the definition of the different levels of our DBD. The information contained in the DBD, as we foresee it currently, belongs to three different levels: the Company DBD (common to every project), the Project DBD (completed through the project progresses) and the Product DBD (containing all information peculiar to the product and exclusively the product). The process to handle the storage of information, from one level to another, needs to be defined in details and the benefits of each level should be identified.

We also want to address the issue of the management of granularity during the different project phases. The levels of detail necessary to describe the architecture of the system and suitable to describe the dynamics of the system are hard to define. Furthermore, this point is likely to impact the SE process of the whole project, the way dependability analyses are performed, and the bridges between SE and dependability.

Finally, addressing the before-mentioned issues should lead us to address the challenges that represent the use of Components Off The Shelf (COTS) in a complex system with high dependability requirements.

5 REFERENCES

Cressent, R., David, P., Idasiak, V. & Kratz, F. 2010. Increasing Reliability of Embedded Systems in a SysML Centered MBSE Process: Application to the LEA Project. *1st M-BED workshop, during DATE 2010,* Dresden, Germany, 12 March 2010.

Cressent, R., Idasiak, V. & Kratz, F. 2011. Mastering safety and reliability in a Model Based process. *Proceedings of the 57th Annual Reliability and Maintainability Symposium, RAMS2011,* Orlando, Florida, USA, 24-27 January 2011.

David, P., Idasiak, V. & Kratz, F. 2010. Reliability study of complex physical systems using SysML. *Journal of Reliability Engineering and System Safety*, Volume 95, Issue 4, April 2010, Pages 431-450.

David, P. & Shawky, M. 2010. Supporting ISO 26262 with SysML, Benefits and Limits. *Proceedings of ESREL 2010,* Rhodes, Grèce, 2010.

Falempin, F. & Serre, L. 2009. French Flight Testing Program LEA Status in 2009. *16th AIAA/DLR/DGLR International Space Planes and Hypersonic Systems and Technologies Conference,* Bremen, Germany, 19-22 October 2009

International Electrotechnical Commission. 1998-2005. IEC 61508. *Functional Safety of Electrical /Electronic /Programmable Electronic Safety-Related Systems. Parts 1 to 7.*

INCOSE, 2004. Systems Engineering Handbook. *International Council on Systems Engineering.* Version 3, 2004.

Langheim, J., Guegan, B., Maillet-Contoz, L., Maaziz, K., Zeppa, G., Philippot, F., Boutin, S., Aboutaleb, H. & David P. 2010. System architecture, tools and modelling for safety critical automotive applications – the R&D project SASHA. *ERTS2 2010, Embedded Real Time Software & Systems,* Toulouse, 19-21 May 2010.

Object Management Group, 2009. Unified Modeling Language. OMG Specification – UML 2.2 Superstructure & UML 2.2 Infrastructure, 2 February 2009.

Object Management Group, 2010. Systems Modeling Language V1.2, June 2010.

Singhoff, F. 2007. The Cheddar AADL Property sets (Release 2.x). *LISyC technical report,* February 2007

Snyder, R., Bocktaels, D. & Feigentaels, X. 2010. Validation fonctionnelle à l'aide d'une transformation SysML/Simulink", *Neptune days N°7*, Toulouse, FRANCE, 18 may 2010, pages 49-53