

# Supporting Change Impact Analysis in System Architecture Design: Towards a Domain-Specific Modeling Method

Afef Awadid<sup>a</sup> and Remi Boyer

*Technological Research Institute SystemX, 91120 Palaiseau, France*

**Keywords:** Domain-Specific Modeling Method, Change Impact Analysis, System Architecture Design, Change Scenario.

**Abstract:** The architecture design update is appraised as a key issue in the domain of cyber-physical systems. One reason for this is its significant cost that can increase up to several folds with each phase of the life cycle, from conceptual design to operation. Therefore, before accepting such update, it is crucial to analyse its impact - the so-called "Change Impact Analysis (CIA)". CIA is based on traceability of dependencies between elements in system artefacts. However, in practice, traceability is not sufficient to allow the automated support of CIA, as it lacks the consideration of CIA parameters. A CIA parameter is any information that could be useful for quantitatively or qualitatively assessing change impact. Against this background, this paper proposes a Domain-Specific Modeling Method for assisting CIA in the context of system architecture design. Such method is composed of two parts. First, a traceability language built upon existing traceability patterns and involving a set of CIA parameters. Second, a modeling procedure referring to a change scenario-based approach that describes how to use such language to provide an automated support for CIA. For the sake of validity, the method is applied in the aeronautical field.

## 1 INTRODUCTION


Many highly complex systems such as cyber-physical systems (e.g. smart grids) have reached a level of complexity that only very few engineers have the breadth of knowledge to gain an even cursory understanding of the system in its entirety. A good architecture design can help overcome this issue by capturing the overall complexity of these systems at a higher abstraction level (Thöne, 2005). Architecture design refers to a sub-process that focuses on "modelling and mapping system functions, structure and predicted behavior of a system" (Eckert and Jankovic, 2016).

System architecture design allows to reduce the system complexity by delineating what one can build, and what requirements are reasonable (Garlan, 2003). Nevertheless, the architecture design of complex systems is appraised as challenging. One reason for this is the need to constantly update such design. Indeed, architecture design update has been identified as a key issue in the cyber-physical systems domain (Wolf and Feron, 2015). Such update is due to different changes such as operating environment,

stakeholder's requirements, and implementation technology. Against this background, it is worthwhile noting that the cost associated with architecture design update can increase up to several folds with each phase of the life cycle, from conceptual design to operation (Chew et al, 2013). Therefore, before accepting any architecture design update, it is crucial to analyse its impact known as "Change Impact Analysis (CIA)".

CIA relies on traceability of relationships between elements in system artefacts (Holt et al, 2016). Nevertheless, a traceability graph is not enough to ensure the automated support of CIA, as it does not explicitly include CIA parameters. The latter refers to any information that could be useful for quantitatively or qualitatively assessing change impact.

To fill this research gap, this paper proposes a Domain-Specific Modeling Method (DSMM) that is made up of two parts. First, a traceability language built upon existing traceability patterns and involving a set of CIA parameters. Second, a modeling procedure defined as a change scenario-based approach that explains how to use such language to

<sup>a</sup> <https://orcid.org/0000-0001-7525-613X>

ensure the automated support of CIA. Given this, the proposed method is an attempt to answer the following research question: how to provide automated support for change impact analysis based on a traceability language?

The rest of the paper is structured as follows. Section 2 introduces the theoretical foundations of this work. The proposed DSMM for supporting change impact analysis in system architecture is presented in Section 3. Section 4 is devoted to the application of the method to a use case in the aeronautic domain. Section 5 rounds off the paper with some future research directions.

## 2 FOUNDATIONS

The key foundations on which this work is built are introduced as follows: change impact analysis (Section 2.1) and domain-specific modeling methods (Section 2.3).

### 2.1 Change Impact Analysis

Change Impact Analysis (CIA) can be defined as the process of exploring the tentative effects of a change in other parts of a system (Kretsou et al, 2021). A typical way to analyze change impact is the following: If I change this requirement or this design element, what are the elements that will be affected and must also be modified? ; If I change this model element, what will the cost be and what effort is required? (Douglass, 2021). In this sense, CIA is much about how change propagates and affects its surroundings (Breivoll, 2010). Building on this, CIA allows to reduce risks, which often arises from changes being made without consideration as to the possible impacts.

When dealing with change in the system architecture design, one generally refers to three

change types: architecture element's addition, alteration, or removal (Damak, 2020). Behind each change type is a Design Change Request (DCR). A DCR can occur for several reasons, instigated from within the design consultancy, the contractor or the project owner/client" (Hindmarch, 2010). For instance, A DCR is issued when the design team found that the design of a subsystem should be changed in order to accommodate constraints from another subsystem (Guegan and Bonnaud, 2018). Such requests undergo reviews in terms of change impact analysis and need to be endorsed before changes get implemented (Akaikine, 2010). These reviews are performed by a change board. For the sake of illustration, the process of CIA as described above is visualized in Figure 1.

In practice, CIA is based on traceability between elements in system artefacts (Holt et al, 2016) (Douglass, 2021). Traceability is defined as "a discernible association among two or more logical entities such as requirements, system elements, verifications, or tasks" (Hunt, 2007). Different types of artefacts can therefore be subject to traceability (e.g. needs, requirements, and models). Although traceability is considered as one of the most frequently adopted technique for CIA (Kretsou et al, 2021), it has been argued that due to the hidden coupling issue and the evolvability of architectural elements, a traceability graph is not sufficient to support the CIA (Yadav et al, 2019). One reason for this is the disregarding of CIA parameters. The latter have been extensively studied in the context of software development (Kretsou et al, 2021) (e.g. change proneness (Jaafar et al, 2014) (i.e. the probability of a software artefact to change) and amount of change (Arisholm et al, 2001) (i.e. the extent of changes that occurred on a software artefact)). However, they are still overlooked in the field of complex systems.

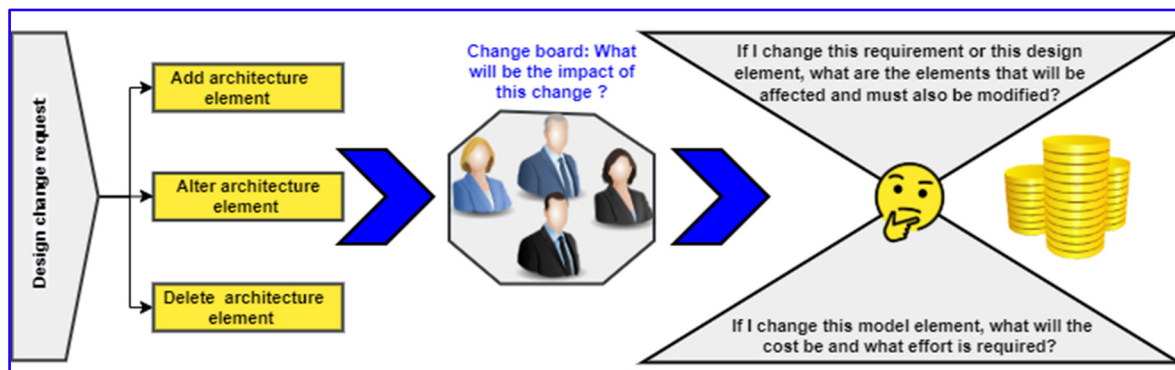


Figure 1: The process of change impact analysis.

## 2.2 Domain-Specific Modeling Methods

Domain Specific Modeling Methods could be a promising candidate for CIA support. This could be ascribed to several reasons. First, in addition to a modeling language, a DSMM provides a modeling procedure, mechanisms and algorithms (Karagiannis and Kühn, 2002). The modelling procedure describes how to use the modeling language in order to achieve results (Awadid and Nurcan, 2019).

Second, a DSMM allows users "to employ familiar concepts to the domain while constructing models of applications" (Hernández et al, 2005) as Domain-Specific Modeling is about creating "models for a specific domain using concepts and terminology from that domain» (Gray et al, 2001). Third, a DSMM "is well suited for domains where the composition of different elements is very flexible" (Leitner et al, 2011), as is the case in the field of system architecture design. Such method is developed by first creating a meta-model that specifies the ontology of the domain and that can be built either from scratch or based on existing solutions (Hernández et al, 2005).

## 3 A DSMM FOR SUPPORTING CIA IN SYSTEM ARCHITECTURE DESIGN

The proposed method involves the domain specific modeling language (Section 3.1)), and the modeling procedure (Section 3.2)).

## 3.1 A Domain-Specific Modeling Language for Supporting CIA

By domain specific modeling language for supporting change impact analysis, we refer to a metamodel that provides the concepts and relationships needed for the construction of traceability models, and that foregrounds the "CIA parameters". Such parameters are required not only to analyze changes, but also to evaluate their potential impacts. As said before, such language can either be established from scratch or adapted from existing language(s) (Section 2.2).

In our case, to specify the intended metamodel, we used two existing traceability patterns coming from the fields of systems engineering and software engineering. The former denotes an ontology definition view showing traceability concepts (Holt et al, 2016), while the latter stands for a traceability pattern for crosscutting (Van Den Berg, 2006). The latter assumes that at least two domains/phases/ levels are somehow related to each other.

The choice of these two patterns drew on our experience gained from several industrial research projects. Once chosen, such patterns are investigated by considering 1) the notion of "CIA parameters", 2) the meaning of the concepts and relationships they use, and 3) the relevance of these concepts and relationships to change impact analysis. Given this, we construct the language/metamodel for supporting CIA. Such metamodel is presented in Figure 2 and lays the basis for the analysis of changes and the assessment of their impacts.

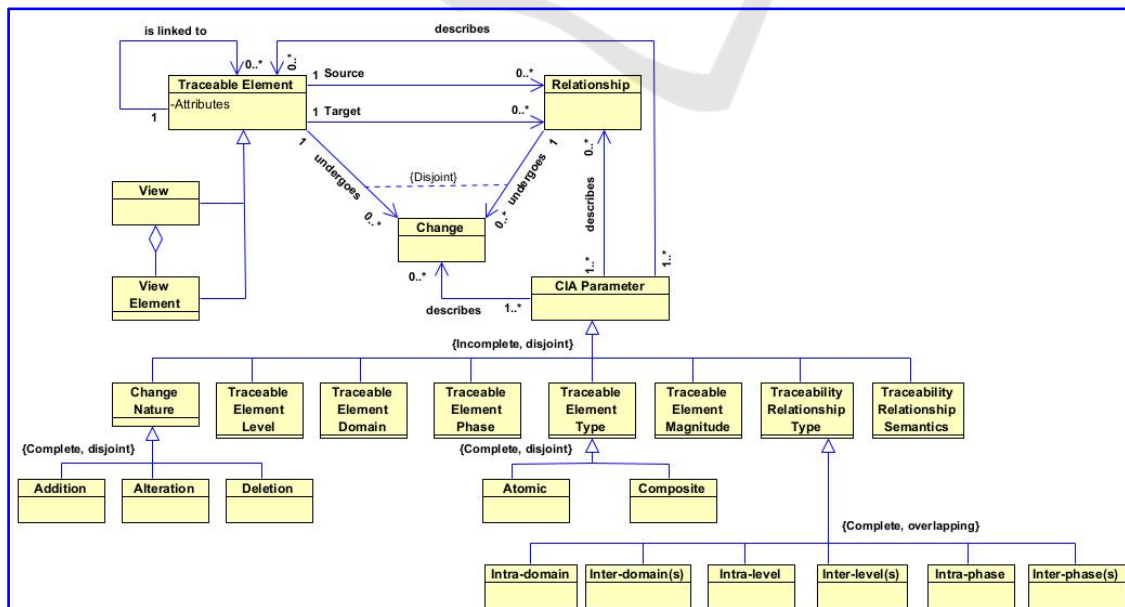


Figure 2: The metamodel for supporting CIA in system architecture design.

Table 1: Metamodel concepts descriptions.

Concept	Description
View	An instance of a viewpoint (i.e. a definition to which a view must conform (e.g. the underlying diagram types)). That instance (view) is made up of View Elements
Traceable Element	The element that can be traced. This may be view or view element. Hence, a traceable element can be abstract. For each two related traceable elements, we distinguish between “Source traceable element” (i.e. the traceable element being modified) and “Target traceable one” (i.e. the traceable element being impacted)
Traceable Element Attribute	A property of a traceable element that could be modified or impacted. Hence, traceable element attribute refers to both the modified attribute(s) of the source traceable element and the impacted attribute(s) of the target traceable one
Relationship	A representation of the actual traceability relationship which is being considered between two traceable elements
Domain	A specific domain of application
Level	Each level is a refinement of the previous one
Phase	A phase can refer to any phase in the system development lifecycle
Change	Any change (in terms of addition, alteration, or deletion) that can be made to a traceable element or a traceability relationship
CIA parameter	Any information that could be useful for quantitatively or qualitatively assessing change impact
Change Nature	Addition, alteration, or deletion (e.g. deletion of an aircraft component)
Traceable element Level	The level of the source traceable element (e.g. the aircraft level)
Traceable Element Domain	The domain of the source traceable element (e.g. the aircraft safety analysis domain)
Traceable Element Phase	The phase of the source traceable element (e.g. the aircraft functional architecture)
Traceable Element Type	It distinguishes between two types of traceable elements: atomic and composite (e.g. an aircraft function is atomic, while aircraft safety requirements are of type composite)
Traceable Element Magnitude	It focuses on how many target traceable elements are related to the considered source traceable element (e.g. there are x target traceable elements that are linked to the aircraft component in question)
Traceability Relationship Type	It is described in terms of domain, phase, and/ or level. We distinguish then between intra-domain (e.g. within the systems engineering domain) and inter-domain (e.g. between systems engineering and safety analysis domains), intra-level (e.g. within the aircraft level) and inter-level (e.g. between aircraft and item levels) , and intra-phase (e.g. within the phase of aircraft functional architecture) and inter-phase relationships (e.g. between the phases of aircraft functional architecture and functional safety verification)
Traceability Relationship Semantics	The meaning of a traceability relationship (e.g. refinement, verification, and validation)

As shown in Figure 2, the resulting metamodel incorporates CIA parameters as inherent constitutive fragment. All the concepts that seem germane to the understanding of this metamodel are summarized in Table 1, where CIA parameters are marked in yellow.

The list of CIA parameters presented in table 1 is not exhaustive. As mentioned earlier, the definition of these parameters are based on a review of existing literature on traceability patterns and on our experience with system architecture design.

### 3.2 A Modeling Procedure: A Change Scenario-Based Approach

As stated previously, a modeling procedure stands for the part of the domain-specific modeling method (DSMM) that explains how the underlying metamodel can be used in order to achieve results. On that basis, the focus of this section is to elucidate the way in which the metamodel presented above can provide support for change impact analysis (CIA).



To do so, we advocate a change scenario-based approach, where the concept of a “change scenario” refers to a description of a given change by instantiating the concepts of the CIA metamodel according to the application domain.

This implies a customized instantiation of such metamodel for the considered application domain. There are at least two motivations that drive the adoption of a scenario-based approach. First, the use of scenarios is highly recommended for evaluations purposes. In this vein, such use has been accentuated as one of the best industrial practices for evaluating architectures (Kazman et al, 2000). Second, a change scenario-based approach is considered as suitable basis for handling analysis complexity (Dobrica and Ovaska, 2011).

The proposed modeling procedure relies on the change-scenario concept and requires the involvement of different experts from the considered application domain. Indeed, two categories of experts are needed. The first category includes experts on system architecture design such as system engineers and system architects. This kind of experts is generally accustomed to architecture design changes. Hence, it is likely to be more willing and able to identify change-scenarios. The second category encompasses experts on system design quality assessment like quality engineers. Such experts are acquainted with impact analysis. Accordingly, they are in the best position to evaluate the severity of change scenarios. Having this in mind, the modeling procedure goes through the following steps:

- Identification of change scenarios according to the application domain. In this step, experts from the first category describe each possible change by instantiating the concepts of the metamodel, including the CIA parameters. It is worthwhile to note here that the concepts that have to be instantiated are determined in accordance with the application domain characteristics. The aim of this step is to point out all the possible changes that can be made to traceability elements or traceability relationships, and that can induce different impacts. Therefore, change analysis can be facilitated. To define change scenarios, the experts resort to previous experience and to existing guidelines and standards in the domain under consideration (for an illustration of this step, see Table 2 and Step 1 in Section 4);
- Given the defined change scenarios, experts belonging to the second category determine the standpoint(s) from which the change impact evaluation can be undertaken. This allows them

to identify the most significant CIA parameters that can be used to quantitatively and/or qualitatively evaluate the impact of each change scenario (for an illustration of this step, see Step 2 in Section 4);

- For each considered standpoint, the same experts define an evaluation criterion along with its scale for scoring. An evaluation criterion refers to either one CIA parameter or the combination of several CIA parameters, among those previously selected in Step 2 (for an illustration of this step, see Step 3 in Section 4);
- Using the defined evaluation criteria, the experts evaluate the impact severity of each change scenario. The evaluated change scenarios are then stored in a tool database in order to be used by the change board, when analysing change requests (cf. Figure 1) (for an illustration of this step, see **Step 4** in Section 4). In light of this, the described modeling procedure needs to be tool-supported, so that new change scenarios can be easily added, and the time expended on evaluating and/ or prioritizing change requests can be significantly reduced.

## 4 A DSMM FOR SUPPORTING CIA: A FIRST APPLICATION IN THE AERONAUTICAL FIELD

The initial evaluation of the proposed DSMM has been realized within the joint academic-industrial research project "S2C-System and Safety Continuity", which aims to support the consistency between model-based systems engineering and model-based safety analysis in the aeronautical industry (De Bossoreille, 2019). The participation of so many experts in this project such as systems, safety and quality engineers makes it a prime candidate for proof-of-concept.

Following the modeling procedure described above, we first identify the two categories of the required experts. Once done, the first category of experts proceed to the identification of change scenarios. For this end, they count not only on their experience, but also on Aerospace Recommended Practices (ARP) viz. ARP4754A (SAE Aerospace, 2010) and ARP4761 (SAE Aerospace, 1996), which allowed them to instantiate only the metamodel concepts that they consider relevant to change analysis. As a result hundreds of change scenarios have been reported. However, due to space limitations, only a small selection of these scenarios is presented in Table 2.

Table 2: A sample of identified change scenarios in the aeronautical field.

Change scenario id	Source traceable element		Target traceable element		Change nature	Traceable element level	Traceable element domain	Traceable element magnitude	Traceability relationship type
	Traceable element	Attribute	Traceable element	Attribute					
1	Aircraft function	Name	FHA_Failure condition	Assumption	Alter	Aircraft	Systems engineering (SE)	=x (i.e. a given number)	Inter-domain & intra-level
2	System function	Description	FHA_Failure condition	Minimal CutSets	Alter	System	SE	<x	Inter-domain & Inter-level
3	Aircraft component	All	Aircraft Safety requirements	Satisfy status	Add	Aircraft	SE	>x	Inter-domain & inter-level
4	System component	All	Safety requirements	Description	Delete	System	SE	<x	Inter-domain & inter-level
5	System component	IDAL	Safety requirements	Assumption	Alter	System	SE	=x	Inter-domain & inter-level
6	Item component	Name	Item_Safety requirements	Description	Alter	Item	SE	<x	Inter-domain & intra-level
7	Aircraft Safety requirements	All	Aircraft functional	FDAL	Add	Aircraft	Safety Analysis (SA)	=x	Inter-domain & intra-level
8	Aircraft Safety requirements	Satisfy status	Aircraft component	IDAL	Add	Aircraft	SA	>x	Inter-domain & inter-levels
9	FHA_Failure condition	probability	Aircraft function	Assumption	Alter	Aircraft	SA	<x	Inter-domains & intra-level

**Step 1:** As illustrated in Table 2, the metamodel concepts that have been instantiated by the experts are: source and target traceable elements, change nature, traceable element level, domain, and magnitude, and finally traceability relationship type. The five last concepts stand for the CIA parameters. Such instantiation is based on the relevance to change analysis and to specific context of the aeronautical field. Against this background, it is interesting to note that the reader not familiar with the terminology used in such field, and hence in Table 2 may refer to Aerospace Recommended Practices (SAE Aerospace, 1996) and (SAE Aerospace, 2010).

**Step 2:** Once change scenarios are identified, the second category of experts determined the standpoint(s) from which the change impact evaluation can be performed. Indeed, two standpoints have been advocated. The first standpoint emphasizes on the impact of the change on the results of systems engineering or safety analysis activities, while the second standpoint concerns the impact with respect to workload related to the resumption of the safety analysis. To evaluate the change impact from these two standpoints, the experts selected the most significant CIA parameters among the five ones instantiated in Step 1 (cf. Table 2). Consequently, only three CIA parameters have been considered: “Traceable Element Level”, “Traceable Element Magnitude”, and “Traceability Relationship Type”.

**Step 3:** The experts specified an evaluation criterion per standpoint. As a matter of fact, the evaluation criterion that has been defined according

to the first standpoint refers to a combination of two CIA parameters viz. “Traceable Element Level” and “Traceable Element Magnitude”. However, that defined according to the second standpoint corresponds to only one CIA parameter: “Traceability Relationship Type”. These two evaluation criteria along with their scale for scoring are given in Table 3 and Table 4 respectively. In both tables, m denotes “minor” impact, me denotes “medium” impact, and M denotes “Major” impact.

Table 3: Evaluation criterion according to the first standpoint.

CIA parameter 1		Traceable element level		
CIA parameter 2		Item	System	Aircraft
		Traceable element magnitude		
	1	m	m	me
	<= x	m	me	M
	> x	me	M	M

Table 4: Evaluation criterion according to the second standpoint.

CIA parameter 3		Rating
Traceability relationship type	Intra-level	m
	Inter-level	me
	Inter-levels	M

Note that “Intra-level” here means that the target traceable element belongs to only one systemic level (i.e. item level, system level, or aircraft level); “Inter-level” implies rather that the target traceable element belongs to two systemic levels, while “Inter-levels” denotes that it belongs to the three systemic levels.

id_scenario	Origin	Entity	Attribute	Nature	Level	Impacted_entity	Severity
1	System	Aircraft Function	All	Add	Aircraft	Assumption	Major
2	System	Aircraft Function	All	Delete	Aircraft	Assumption	Minor
3	System	Aircraft Function	Name	Update	Aircraft	Assumption	Minor
4	System	Aircraft Function	Description	Update	Aircraft	Assumption	Average
5	System	Aircraft Function	Type	Update	Aircraft	Assumption	Major
6	System	Aircraft Function	Function-DAL	Update	Aircraft	Assumption	Average
7	System	Aircraft Function	All	Add	Aircraft	Safety Requirement	Major
8	System	Aircraft Function	All	Delete	Aircraft	Safety Requirement	Minor
9	System	Aircraft Function	Name	Update	Aircraft	Safety Requirement	Minor
10	System	Aircraft Function	Description	Update	Aircraft	Safety Requirement	Average
11	System	Aircraft Function	Type	Update	Aircraft	Safety Requirement	Major
12	System	Aircraft Function	Function-DAL	Update	Aircraft	Safety Requirement	Average
13	System	Aircraft Function	All	Add	Aircraft	Failure Condition	Major
14	System	Aircraft Function	All	Delete	Aircraft	Failure Condition	Minor
15	System	Aircraft Function	Name	Update	Aircraft	Failure Condition	Minor
16	System	Aircraft Function	Description	Update	Aircraft	Failure Condition	Average
17	System	Aircraft Function	Type	Update	Aircraft	Failure Condition	Major
18	System	Aircraft Function	Function-DAL	Update	Aircraft	Failure Condition	Average
19	System	Aircraft Function	All	Add	Aircraft	Effect Classification	Major
20	System	Aircraft Function	All	Delete	Aircraft	Effect Classification	Minor
21	System	Aircraft Function	Name	Update	Aircraft	Effect Classification	Minor
22	System	Aircraft Function	Description	Update	Aircraft	Effect Classification	Average
23	System	Aircraft Function	Type	Update	Aircraft	Effect Classification	Major
24	System	Aircraft Function	Function-DAL	Update	Aircraft	Effect Classification	Average
25	System	Aircraft Function	All	Add	Aircraft	Safety Objective	Major
26	System	Aircraft Function	All	Delete	Aircraft	Safety Objective	Minor
27	System	Aircraft Function	Name	Update	Aircraft	Safety Objective	Minor
28	System	Aircraft Function	Description	Update	Aircraft	Safety Objective	Average
29	System	Aircraft Function	Type	Update	Aircraft	Safety Objective	Major
30	System	Aircraft Function	Function-DAL	Update	Aircraft	Safety Objective	Average
31	System	Allocated Function	All	Add	Aircraft	Safety Requirement	Major
32	System	Allocated Function	All	Delete	Aircraft	Safety Requirement	Minor
33	System	Allocated Function	Name	Update	Aircraft	Safety Requirement	Minor
34	System	Allocated Function	Description	Update	Aircraft	Safety Requirement	Average
35	System	Allocated Function	Type	Update	Aircraft	Safety Requirement	Major
36	System	Allocated Function	Function-DAL	Update	Aircraft	Safety Requirement	Average
37	System	Allocated Function	All	Add	Aircraft	Failure Condition	Major

Figure 3: Illustration of the evaluated stored change scenarios.

To evaluate the impact severity of each change scenario from those identified in Step 1, the experts consider both standpoints, and thus both evaluation criteria, as illustrated in Table 5.

Table 5: Evaluation of change scenarios severity.

Evaluation criterion according to the first standpoint	Evaluation criterion according to the second standpoint	Change scenario severity	
m	m	m	
m		me	m
m		M	me
me		m	me
me		me	me
me		M	me
M		m	me
M		me	M
M		M	M

**Step 4:** Using Table 5, the change scenarios have been evaluated and stored in the tool database for further use by the change board (cf. Figure 3). In this sense, to evaluate a new change request, the change board has just to search for the concerned change within the scenarios database. This avoids having to perform a change impact evaluation from scratch. Figure 3 is offered just for illustrative purposes, as the tool is still under development and some improvements are yet to be done.

## 5 CONCLUSIONS

Architecture design update has been identified as a challenging issue, as it may come at an exuberant cost. Therefore, it is essential to analyse the impact of such update before it is accepted. This is referred to as "Change Impact Analysis (CIA)". The latter rests on traceability of dependencies between elements in system artefacts. However, in practice, traceability is not enough to allow the automated support of CIA, as it does not involve CIA parameters, where a CIA parameter stands for any information that could be useful for quantitatively or qualitatively assessing change impact. To overcome this limitation, the current paper proposes a domain-specific modeling method for supporting change impact analysis in the context of system architecture design. Such method is composed of two parts. First, a traceability language (metamodel) endowed with key CIA parameters. Second, a modeling procedure referring to a change scenario based approach. The method has the potential to be easily extended to cover new CIA parameters.

A first application of the method has been carried out within the S2C research project in the aeronautical field, and was promising. Indeed, it has brought resoundingly positive feedback from industrial partners of the project, who expressed their interest in the change scenario based approach. As future prospects, we plan to perform further validation of the method, and hence to apply it to

several fields such as automotive and marine industries.

## REFERENCES

- Thöne, S. (2005). Dynamic Software Architectures: A Style-Based Modeling and Refinement Technique with Graph Transformations (Doctoral dissertation).
- Eckert, C., & Jankovic, M. (2016). System architecture design. *AI EDAM*, 30(3), 214-216.
- Garlan, D. (2003). Formal modeling and analysis of software architecture: Components, connectors, and events. In *International School on Formal Methods for the Design of Computer, Communication and Software Systems*. Springer, Berlin, Heidelberg.
- Wolf, M., Feron, E. (2015). What don't we know about CPS architectures? In *52nd ACM/EDAC/IEEE Design Automation Conference*. IEEE.
- Chew, K. H., Klemeš, J. J., Alwi, S. R. W., Manan, Z. A. (2013). Industrial implementation issues of total site heat integration. *Applied Thermal Engineering*, 61(1), 17-25.
- Holt, J., Perry, S., Brownsword, M. (2016). *Foundations for model-based systems engineering: from patterns to models*. IET.
- Hernández, F., Bangalore, P., Reilly, K. (2005). Automating the development of scientific applications using domain-specific modeling. In *Proceedings of the second international workshop on Software engineering for high performance computing system applications*.
- Kretsou, M., Arvanitou, E. M., Ampatzoglou, A., Deligiannis, I., Gerogiannis, V. C. (2021). Change impact analysis: A systematic mapping study. *Journal of Systems and Software*, 174, 110892.
- Douglass, B. P. (2021). *Agile Model-Based Systems Engineering Cookbook Improve system development by applying proven recipes for effective agile systems engineering*.
- Breivoll, J (2010) Change Impact Analysis—A Case Study. In *the 8th Conference on Systems Engineering Research*, Hoboken, NJ.
- Damak, Y. (2020). Operational Context-Based Design and Architecting of Autonomous Vehicles (Doctoral dissertation, Université Paris-Saclay).
- Hindmarch, H., Gale, A. W., Harrison, R. E. (2010). A support tool for assessing the impact of design changes during built environment projects. In *2010 IEEE International Conference on Industrial Engineering and Engineering Management*. IEEE.
- Guegan, A., Bonnaud, A. (2018). Assessing the Maturity of Interface Design. In *International Conference on Complex Systems Design & Management*. Springer.
- Akaikine, A. (2010). The impact of software design structure on product maintenance costs and measurement of economic benefits of product redesign (Doctoral dissertation, Massachusetts Institute of Technology).
- Hunt, T. (2007). Vertical and horizontal requirements relationships. Last accessed, 4(22), 2009.
- Yadav, V., Joshi, R. K., & Ling, S. (2019). Evolution traceability roadmap for business processes. In *Proceedings of the 12th innovations on software engineering conference (formerly known as india software engineering conference)*.
- Jaafar, F., Guéhéneuc, Y. G., Hamel, S., Antoniol, G. (2014). Detecting asynchrony and dephase change patterns by mining software repositories. *Journal of Software: Evolution and Process*, 26(1), 77-106.
- Arisholm, E., Sjøberg, D. I., Jørgensen, M. (2001). Assessing the changeability of two object-oriented design alternatives--A controlled experiment. *Empirical Software Engineering*, 6(3), 231-277.
- Karagiannis, D., Kühn, H. (2002). Metamodelling platforms. In *EC-web*.
- Awadid, A., Nurcan, S. (2019). Overlap-Driven Approach for the Conceptualization of Consistency Preserving Modeling Tools. In *25th Americas Conference on Information Systems*.
- Gray, J., Bapty, T., Neema, S., Tuck, J. (2001). Handling crosscutting constraints in domain-specific modeling. *Communications of the ACM*, 44(10), 87-93.
- Leitner, A., Kreiner, C., Mader, R., Steger, C., Weiß, R. (2011). Towards multi-modeling for domain description. In *Proceedings of the 15th International Software Product Line Conference*.
- Van Den Berg, K. (2006). Change impact analysis of crosscutting in software architectural design. In *Workshop on Architecture-Centric Evolution at 20th ECOOP*.
- Kazman, R., Carrière, S. J., Woods, S. G. (2000). Toward a discipline of scenario based architectural engineering. *Annals of software engineering*, 9(1), 5-33.
- Dobrica, L., Ovaska, E. (2011). Analysis of a cross-domain reference architecture using change scenarios. In *Proceedings of the 5th European Conference on Software Architecture*.
- De Bossoreille, X. (2019). Modeling Functional Allocation in AltaRica to Support MBSE/MBSA Consistency. In *Model-Based Safety and Assessment: 6th International Symposium*.
- SAE Aerospace, ARP4754A Guidelines for Development of Civil Aircraft and Systems, Warrendale, 2010.
- SAE Aerospace, ARP4761 Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment, Warrendale, 1996.