Validation, Accreditation or Certification: a New Kind of Diagram to Provide Confidence

Thomas Polacsek*

Abstract

The aim of the argumentation diagram is to organize and visualize, in a synthetic way, all key elements proving the validity of a product's property. The argumentation diagram does not represent the process, but gives the rationale behind all Verification & Validation (V&V) documents. In fact, it lists and organizes necessary evidence in a development life cycle. But the validity of the final assessment requires the validation and the identification of the evidence of each intermediate step. So, in this article, we introduce a generic argumentation pattern and its derivations, whose support a rational organization of all V&V evidence at each step. This pattern stems from legal science and argumentation theory legacies, and it is the basic building block for the argumentation diagram construction.

1. Introduction

During a development life cycle, Verification & Validation (V&V) activities are performed to ensure compliance of the product with expectations about it. These activities are not always limited to internal use, they can also support acceptance activities of the product by a customer. These activities could be acceptance, but also certification or accreditation of the product by what we call an "*authority*" (in this article, we will use the term authority for both customer and accreditation authority). In the context of complex products, the customer may be another department in the same company or a subcontractor. Another example of acceptance can be a review by an experts? committee to upgrade a Technology Readiness Level (TRL) or to pass a milestone project.

To obtain this acceptance, there must be a comprehensive documentation explaining not only results but also input data, assumptions made, techniques applied, etc. Therefore, we need to collect all this documentation and especially to evaluate it. A specific process should structure all these justifications to convince an authority that the product answers to particular questions. An output of this process would be a specific document that must provide, for specified claims, a convincing and valid argument. But, this document, this argumentation, cannot be modelled ex nihilo. It must be built alongside of a development process that accompanies the various stages of V&V and where, for each stage of V&V, argumentation is constructed by aggregating evidence and documents produced at this stage. However, since not all documentations are formal, it seems fruitless to try to establish their validity in a formal way. We are not in a formal world, made only by abstractions, therefore to assess the validity of these justifications we must bend one's efforts towards argumentation.

So, at each step of the product development, V&V documents are created to assert a property of the product or to assert a response to a requirement. For each step, we need to answer the following questions:

- What method was used to assert the property?
- Why is this method relevant?
- What are the restrictions for the use of this method?

*Author version, Validation, Accreditation or Certification: a New Kind of Diagram to Provide Confidence, 459-466, IEEE Tenth International Conference on Research Challenges in Information Science, 1-3 June 2016, Grenoble, France

- On what evidence, or product?s properties, are the uses of this method based for this particular case?

From the answers to these questions, we can articulate a generic step of reasoning, which we will called *generic argumentation pattern*, that explains how, from a method perspective and based on evidence, it is possible to claim a conclusion, in other words, to establish a property based on evidence. From this generic pattern, it will be possible to derive specific instances, *specific argumentation patterns*, for each domain and for each V&V activity.

In practice, we do not want to focus on a single step of reasoning, but establish complex conclusions that involve several steps of reasoning, several applications of our specific argumentation patterns. To do so, we define the *argumentation diagram* which consists to the aggregation of specific instances of our generic argumentation pattern.

The main goal of the argumentation diagram is to provide an overview of all the V&V documentation. Indeed, the argumentation diagram shows clearly, at each step of reasoning, the ins, the outs and evidence in an auditable structure. The argumentation diagram contributes to the understanding, just as well for the authorities, for the review committees or for stakeholders, of the rationale of V&V activities. Having such a diagram allows to navigate in the documents and ease the capacity to find areas where there is a lack of evidence.

Note that there is a strong similarity between requirements engineering and argumentation approaches. In requirements engineering, one of the problems is to refine high level requirements to low level requirements, to break down requirements to achieve into a set of basic requirements. This is a decomposition mechanism, decomposition that yields problems like traceability and completeness. The argumentative approach tries to clearly express how evidence supports a conclusion. The argumentation process follows an aggregation mechanism: it is from the basics that we search to establish a high-level property (property which is in general a response to a high level requirement).

In this paper, we introduce the concepts of argumentation patterns and the argumentation diagram. To design our patterns, we use argumentation theory legacy. Argumentation theory focuses on the links between assumptions and conclusions i.e. on structuring reasoning. The notion of argument obviously refers to the concept of evidence that largely evolved in the history of science and does not have the same meaning depending on whether it is in the formal disciplines, experimental sciences or Humanities. So, in section 2, we present a brief state of the art of argumentation. Then, in section 3, we introduce our generic argumentation pattern and, after that, the concept of specific argumentation patterns. In section 4 we explain how to build an argumentation diagram from specifics patterns. In section 5, we mention how we applied the argumentation diagram to industrial case studies and section 6 is dedicated to the conclusion.

2. State of the art

2.1. It is not valid, it is acceptable

A product, that could be a software, a manufactured good like an airplane, or even a service, is rarely just a product alone, it has an associated industrial development process and it responds to identified needs. A product has features, properties, that meet particular requirements and it is precisely on these properties that we will base our study here. Saying that a product has a given property means that we know that the product has this property. But what is exactly the meaning of *to know*? In the *Theaetetus*, Plato offers various analysis to define the notion of knowledge. In the history of philosophy, the large majority of philosophers use the Plato tripartite theory of knowledge: P is known: (1) if P is true, (2) if we believe that P is true and (3) if we have justifications on that belief. This is called *justified true belief* (JTB).

If we accept the JTB definition of knowledge, the focal point is the concept of justification. Indeed, if one considers that a product

has a particular property, it is reasonable to think that it has it and it is believed that it has it. We evacuate everything linked to fallacies and dishonesty that has no place here. Therefore, following the JTB definition, we need to focus on justifications that guarantee the property. For that, we can look at the work of the argumentation community and the work of some logicians.

If we focus on the justifications of a property, we need to substitute the notion of the validity by the notion of *trust*: trust that all this justifications contribute to some informal proof that a property holds. It is not anymore question of validity, in the sense that the property would be true or false, but the the question to study why a property is acceptable.

There is a long tradition in the study of non-formal statements. Even if the advocate of logical positivism Rudolf Carnap [Car62] said that are non-formal statements vague and incorrect, he speaks of prescientific statement, the logician Charles Hamblin tries to draw a parallel between logical and dialectical arguments. Hamblin studies the false conclusions, and more precisely the relationship between the premises and the conclusion. In [Ham70] he gives a dialectic that allows or prohibits discursive behaviour. In fact, he focuses on fallacies "*an argument appears to be valid but is not*", and he gives a more modern definition of the thirteen types of fallacies listed by Aristotle to which he adds a set of fallacious reasoning that he identified himself. Note that Hamblin claims clearly his preference for the term *acceptability* rather than validity and, like him, in this article we prefer to speak of an *acceptable argumentation* rather than a valid argument.

In the legal field, Perelman and Olbrechts-Tyteca [POT08] define a new theory of argumentation based on a dialectical approach that complements the formal logic. They attack the idea that the only possible proof is formal proof, because if that were the case then it would be impossible to establish the reasoning other than formal reasoning which is "*a wholly unjustified and unwarranted limitation of the domain of our reasoning and proving faculty*". Second, in the same time, Stephen Toulmin [Tou03] defines a model of what is a good argumentation. This model is taught in many American universities to explain the mechanisms of the argument. For instance, it is used in teaching legal argumentation theory or *critical thinking*.

Today, the study of the acceptability of an argument and underlying mechanisms is studied by a variety of disciplines such as computer science (through artificial intelligence), linguistics, epistemology and the legal sciences.

2.2. Toulmin argument model

We will briefly present Toulmin's model of argumentation. As we will see further (section 3), we will use a simplified version of this model to define and structure, in the form of a diagram, justifications underlying a given property. So even if we will not use Toulmin's model strictly speaking, we will give a broad outline. This presentation will allow us to clarify the concepts related to the argument.

In the Toulmin scheme (example Figure 1), any argument is composed of a claim or a conclusion, noted (*C*), "*conclusion whose merits we are seeking to establish*" and facts or data, noted (*D*), "*the facts we appeal to as a foundation for the claim*". In fact, to well-argue a conclusion is to state it by relying on data.

To justify the transition from the data to the conclusion, additional data are used, sometimes implicitly, called warrant. Warrant (*W*) corresponds to the reasoning process and establishes the logical connection between the data and the conclusion. Data and warrant are the support of the argument and distinction between data and warrant is not always easy. Warrants are general rules, they attest to the strength of the argument, while data are facts, evidence. Toulmin adds the concept of backing (*B*) that binds justifications to the warrant. Backing supports the warrant, it is the justification of why the warrant is a reason to accept the claim. Finally, because the claim is not always necessarily true, it is possible to express reservations with modal qualifiers (*Q*).

These qualifiers correspond to concepts such as *possibly* or *probably*. In addition, Toulmin adds to the conclusion the *rebuttal* (*R*), which expresses exceptions: circumstances in which the conclusion is not true.

(*D*) data are encrypted with OpenSSL with a 128 bits key	⟶	*So* (*Q*) certainly (*C*) the information in the file can not be accessed by anyone unauthorized
	Since (*W*) 128-bit key is enough to protect data based on the Advanced Encryption Standard	
	On account of (*B*) Advanced Encryption Standard issued in 2001 recommends a 128-bit key size (FIPS PUB 197)	*Unless* (*R*) someone steals the key

Figure 1: *Toulmin schema example*

For instance, consider the following case (Figure 1): we know the fact that (*D*) "*data are encrypted with OpenSSL with a 128 bits key*". We focus on the argument: if (*D*) is true then we can conclude that (*C*) "*Only authorized persons could read the data*". This inference is based on the Warrant that a 128-bit key is enough to protect data based on the Advanced Encryption Standard, Standard who is the backing here. In addition, in this example the conclusion is not always true. Indeed, the confidentiality of the data is not established in the absolute: an attacker can steal the key. We must therefore add a rebuttal (*R*) "*someone steals the key*".

As we have said before, we are not in a formal logic, we have not an axiomatic system that can give us the value of validity of the logical formula: $D \rightarrow C$. We are here in a rhetorical framework where we try to define if we are dealing with a *good* argument or not.

2.3. Safety cases and other standards

In the context of risk assessment and dependability, a *safety case* is a structured document which provides a set of justifications on whether a system fulfils its safety requirements, a system is acceptably safe (for a given application in a given environment). The UK Ministry of Defence Standard 00-42[1] gives the following definition: "*a reasoned, auditable argument created to support the contention that a defined system will satisfy the requirements*". We find safety cases in many standards, even if the term is not always the same. In the literature, depending on the application domain (defence, digital avionics, automotive industry, nuclear industry and railways), we find: *safety assessment*, *accomplishment summary*, *assurance case*, *certification evidence* or *security case*.

The link between argumentation and safety is clearly done in many standards. So we can quote the Safety Management Requirements for Defence Systems[2]: "*a Safety Case shall consist of a structured argument, supported by a body of evidence*". Furthermore, some standards, like the ISO/IEC 15026[3], offer, without explaining how, to structure a set of assertions and statements (goals and sub-goals) to have an argumentation for high-level claims.

Concerning the representation of safety cases, if the standards say nothing and the first versions of safety cases were a set of documents without any diagram, today the trend is to give a graphical representation of the argumentation. Although there is no real consensus on a safety case graphical representation, there is a strong predominance for Goal Structuring Notation (GSN) in industry. GSN is based on the work of [KW04]. The GSN community tries to develop a coherent approach for the construction, the visualisation, the maintenance and the reuse of dependability argumentation. GSN is

[1] section 4.1, Ministry of Defence, Defence Standard 00-42 Issue 2, Reliability and Maintainability (R&M) Assurance Guidance Part 3 R&M Case, 2003

[2] Def Stan 00-56, p9, section 9.1

[3] ISO/IEC 15026 is a standard for both systems and software. It specifies minimum requirements to support claims in safety, reliability, maintainability, human factors, operability, and security areas.

actually a graphical notation and a set of best practices.

One limitation of GSN is: GSN is fully safety oriented and it was not designed for generic use. The aim of a GSN diagram is to show the demonstration that safety properties are satisfied and risk has been mitigated. Links are made with safety case standards like [Hol15] which shows how to use GSN to express assurance cases in the aeronautical domain (DO-178), but always in safety area. Moreover, still with a safety aim, [SSC+15] shows how GSN might be support healthcare organizations and [GDHKP13] do the same for human-robot interaction.

Because we want to design an argumentation diagram, not only for safety (and, with the extensive literature on GSN, not for safety at all), GSN seems to be too safety tailored for us. Indeed, even if historicaly the notation is based on Toulmin schema, GSN is not anymore related with it [CM14]. For instance, in GSN strategies are decompositional and the approach is clearly top-down. So, we propose to restart from Toulmin's pattern to design a generic, industry oriented, argumentation diagram.

3. Argumentation patterns

The design patterns [AIS77] approach is a way to describe a recurring problem and its associated solution with a high-level of abstraction. A pattern is usually a representation providing a solution considered good for a conceptual problem. The primary aim of the design pattern approach is to build solutions based on experience. The use of patterns is particularly effective when it comes to communicate good engineering practices. Note that there is no single language to express patterns because they are used in a wide range of areas such as software design [GHJV95] or building design [AIS77].

It is in that vein that we will firstly define a generic argument pattern and secondly explain how it is possible, from this generic pattern, to design more specific patterns corresponding to argumentation best practices.

3.1. Our generic argumentation pattern

For our generic model of argumentation, we do not take the Toulmin schema strictly speaking. Indeed, we do not search to define a schema for legal arguments and even less to characterize what arguments are in their philosophical aspects, but to fall within industrial practices. Our generic pattern is the result of trial and error and experimentations conducted in various projects which will be discussed in section 5.

The significant differences between our pattern and the Toulmin schema are:

- We remove the modal qualifiers. For us, a property is acceptable or not, and we are not interested in properties that are "*usually*", "*sometimes*", etc. acceptable. The goal is not that a product satisfies sometimes a property, or its requirements, but to determine if it satisfying it or not. Even for a non deterministic property, like the system "*tossing a coin*", and a property like "*could be head*", we do not want a modal qualifier. Indeed, in this case the qualifier should be included in the property, like "*sometimes head*", and it should not be outside the property. The qualifier is a part of the property, or the requirement, it does not qualifying an existing property;

- for simplicity, we aggregate both warrant and backing in a single concept which we call *rationale*;

- we add the concept of *usage domain* which corresponds to the fact that the pattern is applied in a specific context, which may have to be explained;

- we transform the concept of rebuttal, i.e. specific cases where the conclusion does not apply, by the concept of *restriction* of the conclusion.

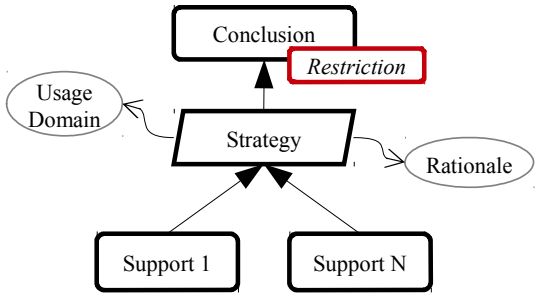

Figure 2: *Generic argumentation pattern*

Our argumentation pattern captures how a property (or a conclusion) is established from a set of supports (or evidence). The goal is to answer the questions presented in the introduction, which are: (1) what method is used, (2) is this method applicable in the context where we are, (3) are there restrictions and (4) on what assumptions are the property based?

We propose a diagrammatic representation of our argumentation scheme (Figure 2). This representation is freely inspired by GSN. This argumentation pattern is based on:

- *Conclusion* (or Property): (unique and necessary)
  It is the property to demonstrate. Generally, the property meets a requirement.

- *Support* (or Evidence): (at least one, necessary)
  Supports on which the conclusion is based. A support can be intrinsic (like data, fact), or refer to another property. If a support is a property, this property is of course the conclusion of another argumentation step.

- *Strategy*: (unique and necessary)
  Method used to determine the conclusion. The Strategy concept corresponds to the reasoning process, or the method used, to establish the connection between the supports and the conclusion. For instance, if the conclusion is "*tests are valid*" then a "*tests coverage*" method could be the strategy, or if the conclusion is "*turbulence model OK*" one could have use the strategy: "*already used in similar projects*".

- *Rationale*: (unique and optional)
  Rationale is the explanation of why the method is applicable here. Justification is binding to the Strategy. It supports the method in argument; it details the justifications for why a method is acceptable. If a strategy is to follow a process defined in a standard, then the strategy is the protocol and rationale is the standard itself.

- *Usage domain*: (unique and optional)
  Usage domain gives the precise conditions of use and the limitations of the method. If we take the example of the application of a standard then the usage domain will describe in what context and for what purposes this standard is applicable.

- *Restriction*: (optional)
  Restrictions are limitations of the conclusion. They are separated from the conclusion because they have no vocation to remain. When we explicitly add a restriction to the conclusion, it is that we think that it will be lifted in the future. An example might be a product restriction that should disappear in a future evolution (evolution which will be the subject of a modification of the argumentation diagram).

3.2. Reflections on support and evidence

Let's have a look more precisely at the support in our pattern. Any demonstration, or proof, is based on pre-established truths. For example, a mathematical demonstration always assumes a set of axioms to be true. These axioms are true by nature, there is no demonstration that proves them, and they are the basic elements of all reasoning in this system. Similarly, any argument is based on a set of postulates, accepted by one that sets out the demonstration as well as its audience.

In the context of legal argumentation theory, [RAR02] defines evidence as: "*data (facts or opinions) presented as proof for an assertion*". The evidence has the particularity to be based on the trust we place in, or the credibility of, those who state them. The acceptance by the audience is not based on evidence itself, but on the trust placed in the one who sets evidence out.

So, for us, supports can be divided in to two categories, first evidence like: results in a scientific paper, piece of information given by an expert, a practice defined in a standard or results given by calculus (like numeric simulation or model checking), second sub-conclusion: a conclusion from another argumentation step.

3.3. Specific argumentation patterns

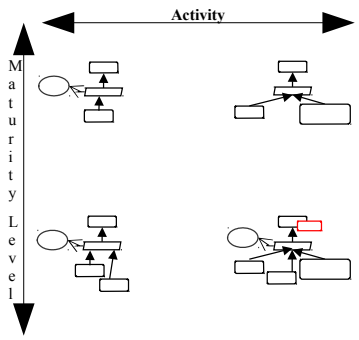

Figure 3: *Specific argumentation patterns: two dimmensions*

The use of our argumentation patterns depends generally on the maturity level and the scope of the product. Therefore, it is impossible to use our generic pattern practically speaking, we need to define specific patterns dedicated to specific uses. These argumentation patterns can be seen in two dimensions: the activity and the maturity level. The activity corresponds to the purpose of the argument, it is the object of the strategy. Thus we define specific patterns for activities like "*validation of an equipment model*" or "*calculation of Pareto-optimal solutions*". Maturity depends on the level expected for the property demonstration. A specific pattern for an activity could be provided in a diversity of forms, some very simple, for less critical activities and/or low maturity level as preliminary design, while other patterns could be very detailed, with a lot of constraints, if they are used in a very strict context like certification.

Of course, a pattern is closely related to the notion of strategy, it corresponds to a particular activity such as a process, the use of a software or the validation by a committee. However, we must do not limit patterns to strategies. In fact, an activity defines a strategy, but also a usage domain, a conclusion, a rationale and a list of mandatory evidence. Obviously, all these elements are generic and their level of detail depends on the maturity level.

Take, for example, a specific pattern for the activity related *the use of results provided by software*. In this pattern, the conclusion is the result given by the software and supports are the information needed by the software to produce its result. Here, we can specify some supports. Indeed, the use of a software is subject to a set of conditions. For instance, a software that guarantees a properties, but only for deterministic systems. Therefore, the set of all supports for the use of this software must include the conditions of use in which the software is used. To check whether these conditions are appropriate, it is necessary to give the usage domain. With these, it is possible to check if the condition of use are included in the usage domain. Note that this verification must be done and, depending of projects, it can be done automatically (if the conditions are expressed formally) or by a human.

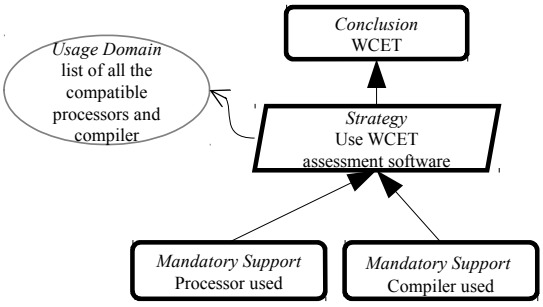

Figure 4: *WCET example*

The pattern for *the use of results provided by software* is still quite generic, it is possible to derive patterns dedicated to each tool. Consider for instance a software to calculate the worst case execution time (WCET)[4] of an application (see a very naive illustration Figure 4). The pattern in this case will have the strategy the use this software, the usage domain will be the list of all the types of processors and compilers compatible with this software and there will be in the supports the type of processor and the compiler really used during the project. Depending on the project, we can add to this pattern rationale, justifications of why the use of this software is allowed.

In a development process, it will be possible to use this pattern (if the software has been used, of course) to support a conclusion as "*maximum time 3 milliseconds*". Suppose the WCET analysis software is based on a formal model of a processor ARM Cortex-A8 (usage domain) and the support in the pattern is the processor a PowerPC G5 970FX, so, in this case, the argumentation no longer holds (because of the non-credibility of WCET software for this processor).

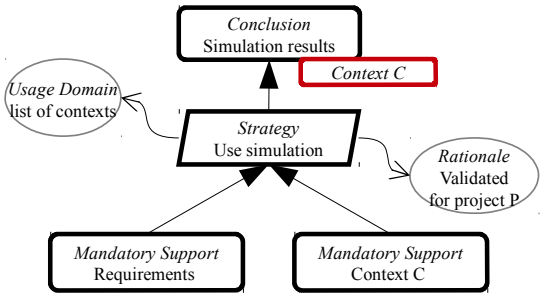

Figure 5: *Simulation example*

Another example of a specific argumentation pattern is *the use of simulation to produce results*. An example of such a pattern is given Figure 5. Here, we have the following necessary supports: (1) a list of simulation requirements, this evidence refers to a document that provides all requirements (such as variables to observe, the conditions to respect, etc.) and (2) the context of the simulation (for avionic simulation it could be the mission and operations considered). We add to this pattern a usage domain that lists all contexts for which the simulation is valid, and a rationale explaining why we can use this simulation. In this example, we chose to limit the rationale to the name of the project for which the simulation is considered valid. Finally, attached to the conclusion is a restriction: the result is true only in the context considered.

3.4. The use of argumentation patterns

One particular goal of our patterns is to be handled by humans. Depending of the project, our patterns can be used: as a checklist of necessary elements to show to an authority "*how to establish*" a property; they can be used to organize and structure all V&V documents in an information system; they can be integrated into a software and linked with the requirements.

In addition, our patterns provide a visual overview of the argumentation step and, even if they are a simplified view, they allow to catch very easily and accurately holding and outs and to check if some pieces of information are missing (such as usage domain or support). Obviously, this is just a visual representation, but, implemented in a software, each box in the diagram that represents a support, usage domain or rationale, should be linked to an object (like a V&V document) in the information system.

One restriction of specific argumentation patterns is reuse in different contexts. Specific patterns are very business oriented, and it is very difficult to use it from a project to another, unless the two projects have the same activities. But, argumentation patterns are useful because once they are defined for specific activities, they are established once and for all. Thus, they guide and give a framework both to establish the justifications of a property and to audit these justifications.

[4]Worst-Case Execution Time, the maximum execution time of a task for a given hardware platform

4. Argumentation diagram

As for the Toulmin schema, our patterns correspond to a reasoning step, i.e. how, from facts, it is possible to infer a conclusion. In practice, we obviously not want to focus on a single step of reasoning, but establish complex conclusion that involve several steps of reasoning. Moreover, all these reasoning steps taken together form what we wall an argumentation diagram.

An argumentation diagram is a diagram that captures the rational logical structure of all evidence that leads to accept a high-level property. In an argumentation diagram, on the top there is a conclusion and, on the bottom, the leaves are evidence (data, fact, etc.). However, an argument diagram is just a representation of how the set of elements that lead to a conclusion are structured. In order to have more V&V details, the diagram must refer to document(s) in the information system.

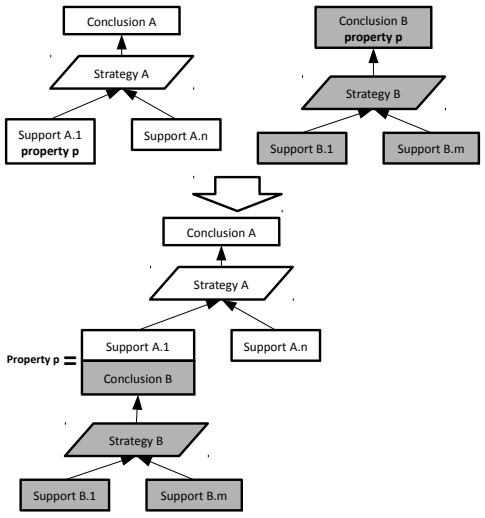

Figure 6: *Argumentation step chaining mechanism*

We build the argumentation diagram from the specific argumentation patterns. For each reasoning step, we instantiate a specific argumentation pattern and the conclusion of this pattern becomes a support for the next step (see Figure 6). In fact, For building an argumentation diagram we used the argumentation step chaining mechanism: one conclusion of a level becomes a support for the next level.

The step chaining mechanism needs to be used carefully. Firstly, the conclusion of one level could be close to the next level's support: close, but not necessarily exactly the same. Secondly, two patterns instantiated could be incompatible for chaining. Indeed, both usage domains of each, or both rationales, could be mutual exclusive.

An effect of chaining two reasoning steps is that, in some cases, restrictions can be removed. Indeed, a conclusion could be restricted in a level and, if some conditions are fulfilled, it is removed to the next level. Take for example, a conclusion "*the equipment is validated, but only below* 70° *degrees*". We have two choices: it is possible to take all the sentence as conclusion or it is possible to split it in two parts, one is the conclusion and the other is the restriction "*below* 70° *degree*". If in the next argumentation step it is established that the system, and therefore the equipment, always operates below 70°, the use of a restriction is more clever. Indeed, this case, we have the same conclusion in the two steps, but, in the second argumentation step, the restriction is released (see Figure 7).

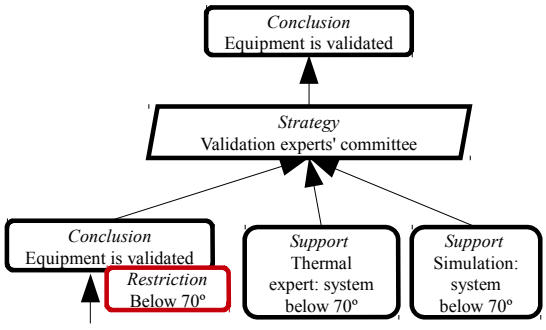

Figure 7: *Restriction release example*

5. Case Studies

We applied successfully our argumentation patterns methodology in the European project Thermal Overall Integrated Conception of Aircraft[5] (TOICA). In TOICA, the partners develop

[5] http://www.toica-fp7.eu/

a big simulation named the Global Thermal Aircraft (GTA). Naively, we can see this simulation as an aggregation of simulations. The aim of the GTA is to simulate the aircraft thermal behaviour. Results of the GTA are key inputs for decision making like structural material choice, identification of particular thermal risks, equipment qualification or design trade-off. Consequently, there is a need for a strong confidence in the GTA results and we designed, with thermal experts, an argumentation diagram to justify to the decision makers why they can trust the simulation results. We defined specific argumentation patterns dedicated to simulation context and a preliminary integration of the argumentation diagram in the information system was performed with other partners.

GTA results are used in an expertsŠ committee and it appears that the argumentation diagram supports the experts during the reviews. It structures the documentation provided as evidence of the trust in the simulation results, then to check this structure against identified argumentation patterns that will help identifying lacks or misuse of elements. Another advantage of using argumentation diagram is to cope with what we call the tsunami of documents. Today, every simulation process produces many documents and experts and architects need to make sense of it. By giving a global picture, a graphical representation of the rationale, the argumentation diagram could support the architect to deal with the tsunami of V&V documents.

We also applied argumentation pattern in the context of MIMOSA project [BBD+16]. In MIMOSA, a certification process for embedded avionics software was defined in close cooperation with certification authorities. This process is based on the concepts of argumentation, model-driven engineering and a component-based software engineering. The goal of the certification is to have the compliance of a system with some requirements approved by an authority. In this project, the system was a part of the avionics software that should, like the rest of the aircraft, be authorized to fly (the certification). During MIMOSA, we defined with the help of experts, specific patterns for real-time embedded software validation activities. The purpose of these patterns was to enable the certification authorities to find very easily the lack or misuse of evidence. In addition, we have shown how, using the argumentation diagram, it was possible to structure the documentation provided to the authority.

To conclude, today, in the context of TOICA, we start a new activity. Aircraft manufacturers have a process to assess the workload for a new aircraft design. In this process, are involved: software, experts' analysis, assumptions, etc. We currently design specific argumentation patterns dedicated to build the confidence in the results given by this process.

6. Conclusion

In this article, we show how the question of validity of a high-level claim for a product must be substituted by the acceptability question. Based on existing work, and more specifically on ToulminŠs work, we laid the foundation for what should be a "*good*" argumentation, good in the sense of well-formed and auditable. The idea of structuring V&V elements in the form of a diagram makes its way mainly through concepts such as Assurance Case, but it is essential that the argumentation diagram is not limited to boxes and arrows and it should be built based on work conducted in linguistics and law. With this in mind, future work should take into account advances in argumentation theory. For instance, Douglas Walton [Wal96] defined patterns of critical questions for experts' testimonies. On this model, we could define sets of questions attached to each argumentation pattern, these questions could support the review of the argumentation diagram or could contribute to support an engineer to understand how to instantiate a specific argumentation pattern.

Acknowledgment

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n°604981.

References

[AIS77] Christopher Alexander, Sara Ishikawa, and Murray Silverstein. *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press, New York, August 1977.

[BBD+16] Pierre Bieber, Frédéric Boniol, Guy Durrieu, Olivier Poitou, Thomas Polacsek, Virginie Wiels, and Ghilaine Martinez. MIMOSA: Towards a model driven certification process. In *Proc. 8th Int. Congress on Embedded Real Time Software and Systems (ERTS'16)*, 2016.

[Car62] R. Carnap. *Logical Foundations of Probability*. University of Chicago Press, 1962.

[CM14] Valentin Cassano and T. S. E. Maibaum. The definition and assessment of a safety argument. In *25th IEEE International Symposium on Software Reliability Engineering Workshops, ISSRE Workshops*, pages 180–185. IEEE Computer Society, 2014.

[GDHKP13] Jérémie Guiochet, Quynh Anh Do Hoang, Mohamed Kaâniche, and David Powell. Model-Based Safety Analysis of Human-Robot Interactions: the MIRAS Walking Assistance Robot. In *International Conference on Rehabilitation Robotics (ICORR)*, pages 1–7, Seattle, United States, 2013.

[GHJV95] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-oriented Software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.

[Ham70] C.L. Hamblin. *Fallacies*. University paperback. Methuen, 1970.

[Hol15] C. Michael Holloway. Explicate '78: Uncovering the implicit assurance case in do-178c. In *23rd Safety-Critical Systems Club (SCSC) Annual Symposium*, February 2015.

[KW04] Tim Kelly and Rob Weaver. The goal structuring notation /- a safety argument notation. In *Proc. of Dependable Systems and Networks 2004 Workshop on Assurance Cases*, 2004.

[POT08] C. Perelman and L. Olbrechts-Tyteca. *Traité de l'argumentation: La nouvelle rhétorique*. UBlire - Fondamentaux. Éditions de l' Université de Bruxelles, 2008.

[RAR02] J. Lynn Reynolds Rodney A. Reynolds. *Evidence*, pages 427–446. SAGE Publications, Inc., 0 edition, 2002.

[SSC+15] Mark-Alexander Sujan, Peter Spurgeon, Matthew Cooke, Andy Weale, Philip Debenham, and Steve Cross. The development of safety cases for healthcare services: Practical experiences, opportunities and challenges. *Rel. Eng. & Sys. Safety*, 140:200–207, 2015.

[Tou03] Stephen E. Toulmin. *The Uses of Argument*. Cambridge University Press, Cambridge, UK, 2003. Updated Edition, first published in 1958.

[Wal96] Douglas N. Walton. Practical reasoning and the structure of fear appeal arguments. *Philosophy and Rhetoric*, 29(4):301–313, 1996.