A Model-based Approach to Certification of Adaptive MILS

Dorien Koelemeijer
Rasma Araby
Atsec – Svardvagen 3C, 18233
Danderyd, Sweden
dorien.koelemeijer@atsec.com
rasma.araby@atsec.com

Ayoub Nouri
Marius Bozga
Univ. Grenoble Alpes, CNRS,
Grenoble INP* – 700 avenue centrale,
38401 Saint Martin d'Hères, France
ayoub.nouri@univ-grenoble-alpes.fr
marius.bozga@univ-grenoble-alpes.fr

Rance DeLong
The Open Group – Apex Plaza,
Forbury Road Reading,
Berkshire RG1 1AX, UK
r.delong@opengroup.org

ABSTRACT

In this work, we tackle the problem of certifying Adaptive systems. These are able to automatically perform self-reconfiguration at runtime, which makes classical certification approaches inapplicable. The need for certification approaches for these systems is thus becoming urgent, especially due to their prevalent use in safety- and mission critical settings. Due to the inherent complexity of adaptive systems and the absence of a principled methodology for their construction and assurance, there has been little movement by certification authorities to accept such systems. Among the challenges for certification are a way of generating an adequate assurance case for initial state of the adaptive system and for each step in its incremental adaptation, and generation and management of the evidence upon which the assurance case relies. We contribute in this research by proposing a novel modular approach to the certification of adaptive systems in the context of the Adaptive MILS architecture. The proposed approach is backed by an Evidential-Tool Bus implementation that allows a continuous on-demand generation of assurance cases.

KEYWORDS

modular assurance cases, evidential tool-bus, adaptive MILS, dynamic reconfiguration

ACM Reference Format:
Dorien Koelemeijer, Rasma Araby, Ayoub Nouri, Marius Bozga, and Rance DeLong. 2018. A Model-based Approach to Certification of Adaptive MILS. In *Proceedings of 4th International Workshop on MILS: Architecture and Assurance for Secure Systems (MILS-Workshop 2018).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

As the world becomes increasingly interconnected, there is a growing reliance on Internet of Things (IoT) technology, autonomous systems, cloud computing for safety- and security-critical applications, and other adaptive systems (i.e. systems that change their behaviour at runtime to maintain viable operation of the system in the face of environmental changes or failures within the system). Moreover, countries rely on the correct functioning of adaptive systems to support critical infrastructures, which include energy and water supply, transportation infrastructure and financial and healthcare systems [20] [6].

The benefits of adaptive systems over fixed design systems include their capability to engage in high numbers of interactions without requiring action by another system, or human interaction, to change their state or behaviour [1]. This additionally means that the system is no longer susceptible to human fallibility, and a decrease in errors is therefore expected. In addition, adaptive systems are capable of performing any non-linear mapping between input and output patterns, and have the capacity for adaptive learning, thereby allowing them to adapt to new inputs [13].

Adaptive systems demand high assurance with respect to reliability, robustness and resilience to ensure their dependability. To ensure critical adaptive systems are fail-safe, an evaluation methodology is required to review, at runtime, whether the system operates correctly according to its safety and security policy. Such an evaluation methodology therefore requires capabilities to evaluate dynamic architectures in a timely manner. Traditional and standards-based evaluation methodologies do not provide the capabilities to evaluate or certify adaptive systems, since they are solely able to review the static architecture of a system against a set of defined requirements, as opposed to dynamic architectures. Moreover, traditional evaluation methods lack the abilities to review systems at runtime and transitions between system configurations, which are essential to adequately verify the safety and security of systems that constantly self-adapt [2].

In the context of the CITADEL project [5], we develop a specific vision of Adaptive MILS[1] systems as illustrated in Figure 1. CITADEL represents a conservative extension to static MILS that maintains key imperatives upon which MILS is founded: viz. the provision of sound methodology for design, and an assurable platform for deployment, of systems that require high assurance of key system characteristics; and, accompanying theory, techniques and tools to achieve and to demonstrate the needed assurances. According to this vision MILS systems are organized into a collection of inter-dependent planes, providing foundational, operational and adaptation support. More precisely, adaptation capabilities are further refined into different planes following a sense-process-actuate

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 700665 CITADEL - Critical Infrastructure Protection using Adaptive MILS.

MILS-Workshop 2018, June 2018, Luxembourg
2018. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

[1] "MILS" is a design paradigm for safety- and security-critical systems composed from independently developed high-assurance components. MILS components are required to conform to a coherent set of specifications under development. Authors of these specifications, vendors of MILS components, and the architects of MILS systems face two challenges unfamiliar to the developers of traditional software and systems: *assured composition* and *substantiated claims*.

scheme, namely monitoring, adaptation and (re-) configuration. Foundational and operational planes represent respectively the low-level platfom (which provides guarantees such as separation kernels and networking) and applications running on top of it. Within the CITADEL framework, certification is a transverse plane that underpins all the other ones.

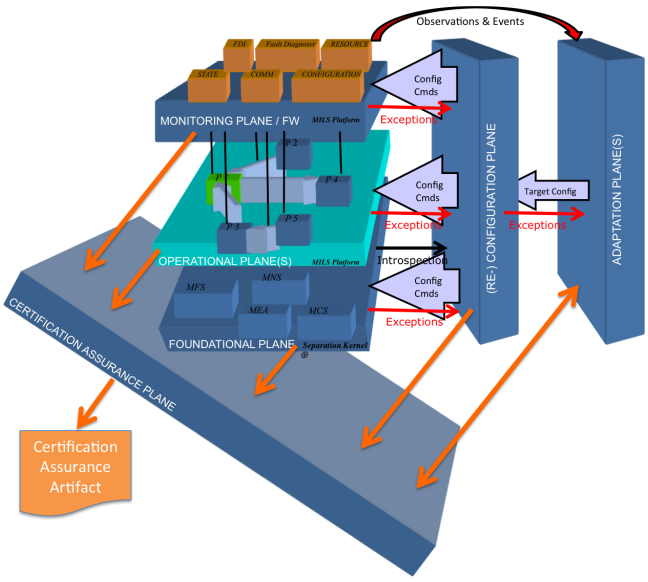

Figure 1: Vision for Adaptive-MILS in CITADEL

This research therefore suggests a certification strategy for Adaptive MILS systems that satisfies the previous imperatives by providing a (meta-)model for reconfigurable systems, a modeling language, analysis tools, a monitoring framework, adaptation and reconfiguration subsystems, a dynamic implementation platform, and a certification assurance subsystem that maintains an up-to-date, consistent, and always-available *assurance case*. An assurance case is the embodiment of the presentation of goals or claims, and the supporting evidence. The validation and certification strategy proposed in this research provides opportunities to on-demand generate and update existing assurance cases, which are supported by evidence that yields proof of the system's specified properties. In accord, different techniques and mechanisms are needed to generate the assurance case and manage the evidence needed for certification during system operation.

The *evidential tool bus* (ETB), a concept first introduced by Rushby [19], provides a framework for propagating system-level and inter-component constraints, enforcing standards, checking consistency, and providing automated review of evidence as a proxy for certification experts. A prototype of the ETB concept has been demonstrated by SRI International [8]. This paper proposes an implementation of the concept of ETB for the construction and continuous review of Adaptive MILS assurance cases, the "AM-ETB". It introduces the notion of assurance cases patterns and proposes design workflows for their instantiation with respect to Adaptive MILS system model [3]. A core part of AM-ETB implementation consists on evidence production (through well defined interfaces with external tools/operators) and management that take into account evidences context.

Previous efforts have developed techniques and tools for creating assurance cases for systems that must undergo rigorous scrutiny under certification regimes for safety and security. The authors of the Distributed MILS for Dependable Information and Communication Infrastructures (D-MILS) [11] project, have made advances in this area, in particular automated linkage between the modelling and verification framework and the assurance framework. That work covered static architectures and configurations, which constructed and configured before deployment and that are static at runtime. Adaptive MILS therefore progresses the D-MILS project by providing an approach to certification of adaptive systems, by enabling the evaluation of dynamic architectures in real time.

The remainder of the paper is organized as follows. In Section 2 the architecture of a modular assurance case for Adaptive MILS is described, which is supported by the employment of a set of assurance case argument patterns. In Section 3, the capabilities of the AM-ETB with regard to continuous assurance case generation and validation are outlined. Section 4 provides insight into the requirements for the validation and certification of Adaptive MILS. Finally, Section 5 presents a summary of the approach to certification of Adaptive MILS proposed in this paper.

2 ASSURANCE CASES FOR ADAPTIVE MILS

An assurance case uses a structured set of arguments and a corresponding body of evidence to demonstrate that a system satisfies specific claims with respect to its security and/or safety properties [15]. The following sub-sections will grant insight into modular assurance cases for Adaptive MILS systems, and the employment of assurance case argument patterns to enhance the process of constructing assurance cases. The approach is clarified on the basis of an illustration of the Adaptive MILS foundational plane assurance case argument pattern.

2.1 Modular Assurance Cases

Assurance cases for Adaptive MILS systems are modular, allowing for both top-down as well as bottom-up approaches to evaluation. This means that the central argument claim demonstrates that the compositional behaviour of separate components will together meet the security or safety policy [10]. This is in line with what Rushby proposes: "The idea of a structured argument is to facilitate modular comprehension and assessment of the case. If we look at this top-down, we divide each claim into components whose conjunction implies the claim, and recurse down to subclaims supported by evidence; if we look at it bottom-up, we treat each evidentially-suported subclaim as an independently settled fact and conjoin these to produce higher-level subclaims that combine recursively to deliver the top claim" [18]. To demonstrate that the compositional behaviour of the separate components meets the high-level security or safety policy, the separate components are required to satisfy their local policy. A local policy encompasses safety or security requirements defined along the Adaptive MILS model.

The structure of a modular assurance case of an Adaptive MILS system can be regarded in Figure 2. The System Properties argument comprises a top claim supported by the six distinctive planes

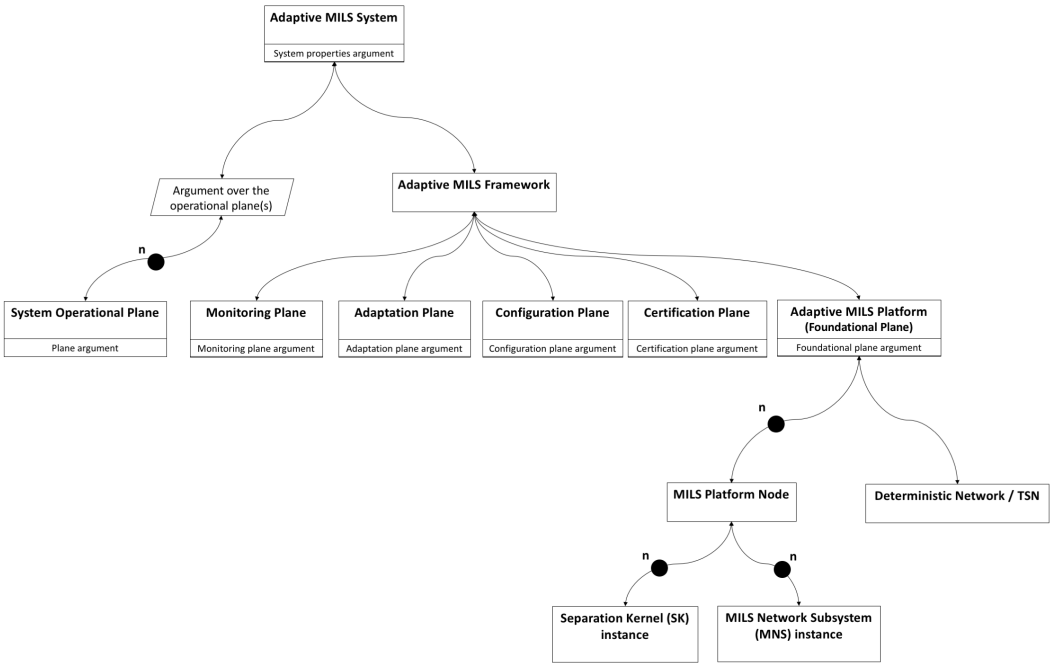

Figure 2: Modular Assurance Cases for Adaptive MILS

descibed earlier. These are divided into the arguments over the operational plane(s) and the Adaptive MILS framework that includes the remaining planes. The local policies of the various planes together are required to be met in order to satisfy the security and/or safety policy of the Adaptive MILS system. For instrance, the Adaptive MILS Platform (the foundational plane), which supports the Adaptive MILS Framework, comprises arguments over the underlying deterministic network and the nodes of the MILS platform. Accordingly, the goal of MILS platform nodes and the goal of the Net (Deterministic Network/TSN) together support the local policy of the Adaptive MILS platform. The MILS Platform Node goal is supported by the goals of the SK instance(s) and MNS instance(s).

To enhance the flexibility of the validation and certification of Adaptive MILS, the construction of assurance cases relies on a set of *assurance case argument patterns*. The following subsection will grant more insight into the benefits and application of the assurance case argument patterns (henceforth referred to as argument patterns) to establish assurance cases for Adaptive MILS.

2.2 Assurance Case Argument Patterns

Argument patterns define the structure of arguments in the modular assurance case for Adaptive MILS systems, and are used to simplify the establishment and review of these assurance cases [21]. The argument patterns specify the requirements to instantiate the claims, and the evidence to support these claims for certain (safety or security critical) components of Adaptive MILS. When constructing an assurance case for a system, the argument patterns are instantiated with information concerning the design, development, analysis and verification of the system. Figure 3 provides a depiction of the hierarchy of the argument patterns that may be used to build the Adaptive MILS assurance case.

Argument patterns for each of the planes of the Adaptive MILS framework are established, since the planes are largely static (i.e. the planes usually comprise the same set of components). The plane argument patterns contain components, and compositions of components. A composition argument concerns a group of components, and, accordingly, contains one or several arguments over these components. Each composition argument may additionally include one or multiple composition arguments (i.e. a composition can include other compositions). Arguments over the separate components of an Adaptive MILS system can be established by implementing the process argument pattern [9].

To verify that the components in an Adaptive MILS system satisfy the local policy, evidence is required to be gathered and assessed. Evidence leaf nodes are therefore included in the process argument pattern. However, as not only components may provide evidence, additional evidence nodes can be incorporated in the assurance case by adding these to the appropriate arguments when instantiating the respective argument patterns. Moreover, components may have additional properties such as person, organisation, artefact, technique and trusted component arguments associated with them, which are also capable of providing evidence to support the claims made in the assurance case.

2.3 Illustration on the Foundational Plane

The employment of modular assurance cases, used for the validation and certification of Adaptive MILS, is elaborated on in this

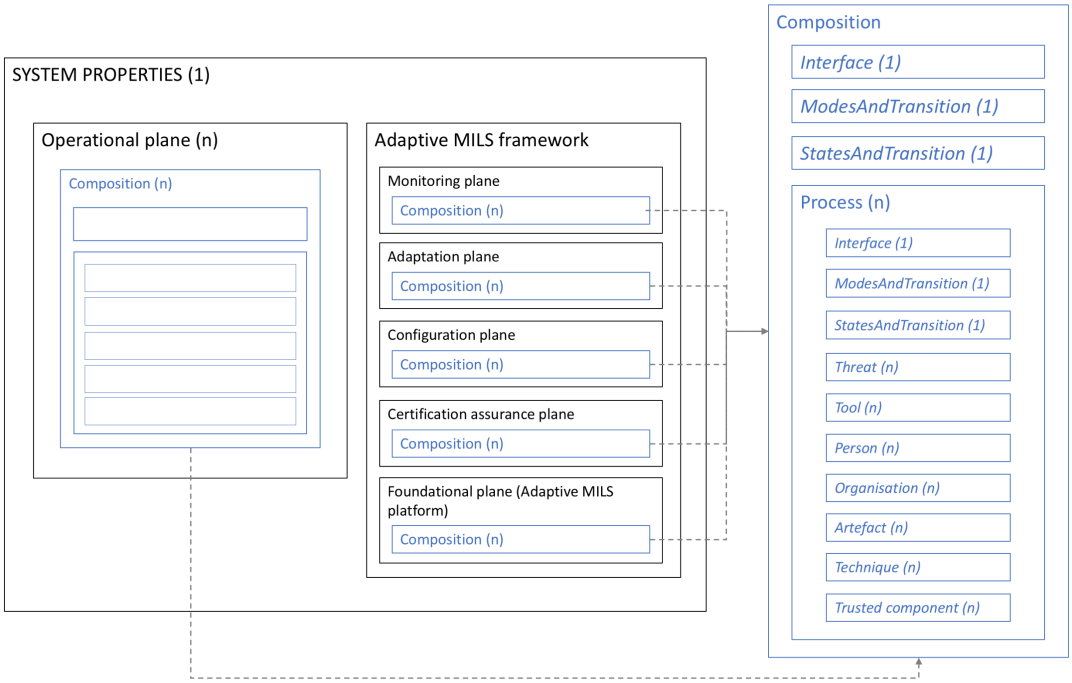

Figure 3: Assurance case argument pattern hierarchy for adaptive MILS. The composition pattern structure on the right side is an enlargement of the composition pattern illustrated on the left side.

section on the basis of an illustration of the Adaptive MILS argument pattern for the foundational plane (Figure 4). The foundational plane argument patterns, similar to all Adaptive MILS argument patterns, are represented in the Goal Structuring Notation (GSN). GSN assurance cases are constructed and visualised by a set of GSN elements that collectively establish a goal structure, which allows for a comprehensible structure of documentation, and ensures a clear mapping between evidence and claims in the arguments of the assurance case [12]. The various GSN elements are additionally explained in the remainder of this section.

The root node of the foundational plane argument pattern encompasses a *goal element*, which is used to specify a claim. A *context element* is attached to the goal element, which refers to the Adaptive MILS system model, and may specify the local policies or other requirements necessary to support the goal or sub-goal [12]. The goal of the foundational plane is supported by various foundational components; the platform node(s), containing separation kernel instances and MILS Network Subsystem (MNS) instances, a Network Scheduling Master (NSM) and the Time Sensitive Network (TSN) [14].

Firstly, a *strategy element* ("Strategy:platformNode") is included to make an argument over the desired behaviour and requirements of each MILS platform node. Strategy elements in GSN are utilised to make an argument about how sub-goals and evidence support the goals, and are displayed as parallelograms. The strategy element is followed by a *sub-goal element* ("Goal:platformNode"), which may consist of safety or security requirements of the system [12]. In this case, the sub-goal is employed to ensure each platform node meets the requirements stated in the local policy of the foundational plane. The sub-goal is followed by two strategy elements, one of which ("Strategy:mnsInstance") is followed by an *undeveloped goal element*, which is depicted as a goal element with a diamond symbol at the centre-bottom and presents a claim which is intentionally left undeveloped in the argument [12].

The other strategy element ("Strategy:separationKernel") comprises an argument over the desired behaviour and requirements over each separation kernel instance. The separation kernel instances enable the system to run applications of diverse security domains on the same processor. The sub-goals following the strategy over the behaviour and requirements of the separation kernel are eventually followed by *evidence elements*. Evidence elements comprise the evidence to support the claims, and are directly connected to their parent sub-goals without intervening strategies. In this case, the evidence to substantiate the sub-goals of the argument consists of a CC evaluation that is compliant to the EURO-MILS protection profile [7].

Secondly, an *away goal element* ("Module:interface") is attached, which indicates that the claim is represented in another argument in the assurance case [12]. In this case, the away goal ensures that the interaction between components and compositions of components of the plane satisfies the security architecture, and is supported by the interface argument pattern.

The NSM monitors and adapts the network when configuration change is necessary. The goal of the NSM in the argument pattern of the foundational plane encompasses an undeveloped goal element.

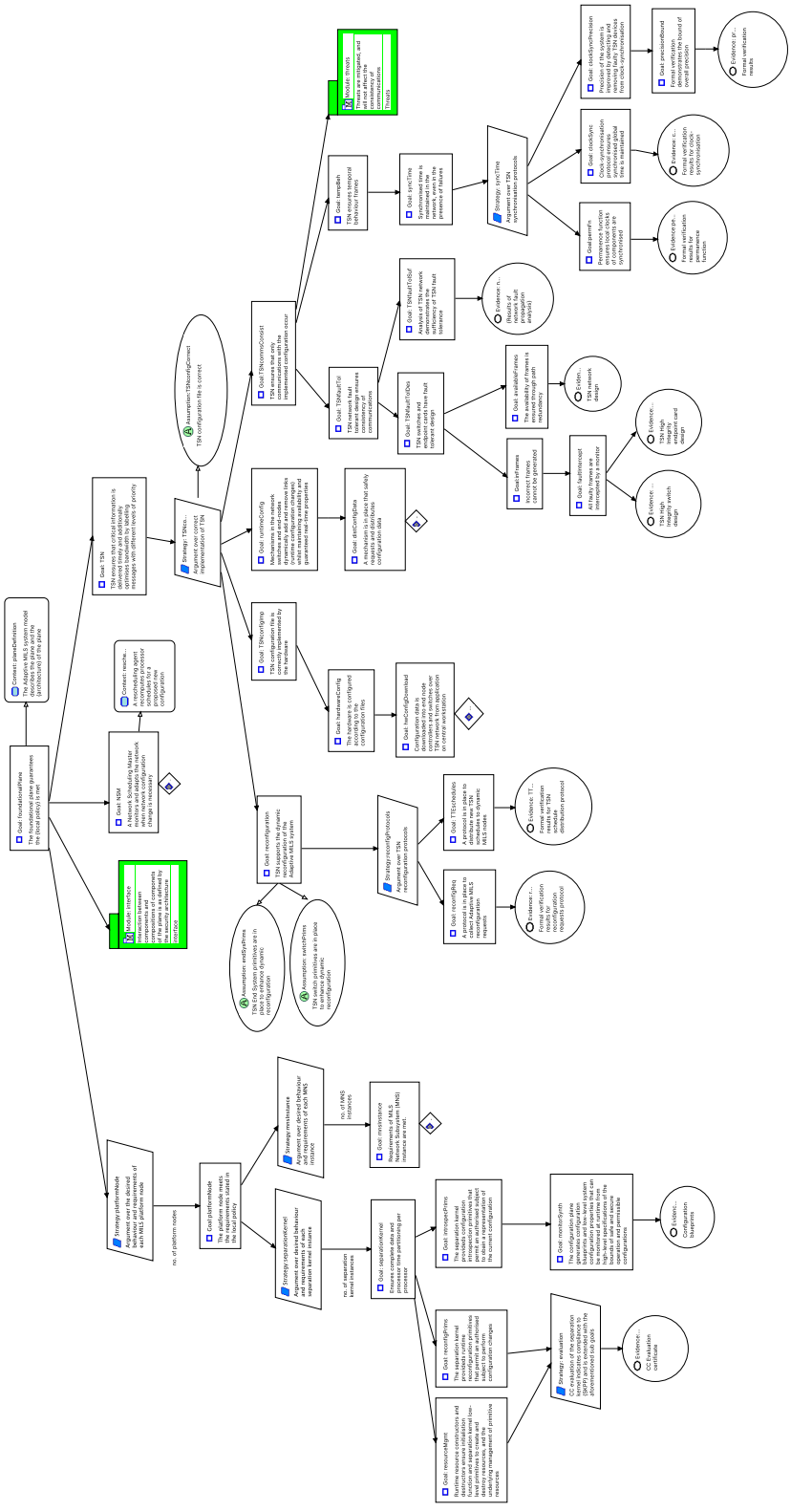

Figure 4: Foundational plane argument pattern

Finally, on the right side of the argument pattern, a sub-claim is included for TSN ("Goal:TSN"), which enhances communication over the Adaptive MILS by taking into account that applications of mixed time criticality share a single network. The goal is followed by a strategy element, to which an *assumption element* is attached [12]. The assumption element signifies the assumptions under which the system or design satisfies the claims(s). Assumptions can be used to support goals, sub-goals or strategies. It is important to note that all sub-goals, supporting the goal in the root node, eventually are supported by either evidence elements, undeveloped goals (that are developed at a later time to include evidence) or away goal elements. Thus, all arguments in the assurance case are ultimately supported by evidence.

3 AM-ETB: AN EVIDENTIAL TOOL BUS FOR ADAPTIVE MILS

AM-ETB is developed as part of the Certification Plane to provide automated support for continuous assurance case generation and evidence checking for Adaptive MILS. Two major usages are foreseen for AM-ETB, respectively

- *apriori* construction, when the assurance case and evidence is constructed for a *future* system configuration, usually before an adaptation/reconfiguration step is applied on the system.
- *just-in-time* construction, when the assurance case and evidence is re-constructed for the *currently running* configuration of the system.

We emphasize that, in both cases, the evidence covers two orthogonal aspects: (1) the correct operation of the system in some stable configuration (resp. current, future) and (2) the correct reconfiguration, that is, its operation during the transient phase, when moving from one stable configuration (resp. previous, current) to the next one.

From an operational perspective, the AM-ETB includes a workflow sequencing system that manages the performance of activities, as part of a *workflow process*, that results in an *assurance case*. Activities are carried out by agents and/or tools that perform reasoning or verification tasks. Results of the activities contribute to establishing higher level goals and/or the chain of reasoning expressed by assurance case patterns. The AM-ETB gathers the results of activities as *evidence* into a database and records the logic of the combination of the evidence to support the certification and assurance goals.

Figure 5 presents the overall architecture of the AM-ETB. The AM-ETB comprises a core workflow engine, a database including assurance case patterns and evidence, and interface agents to run external analysis and verification tools.

3.1 AM-ETB Functional Description

We introduce hereafter the key components and functionalities of AM-ETB. More details are available in [4].

Assurance Case Argument Patterns. Assurance case patterns represent generic assurance cases constructed for specific goals and categories of systems and/or system components. Specific top goals include both the nominal operation and the reconfiguration of the system. Assurance case patterns are intended to drive the execution of the AM-ETB. That is, AM-ETB instantiates the pattern according

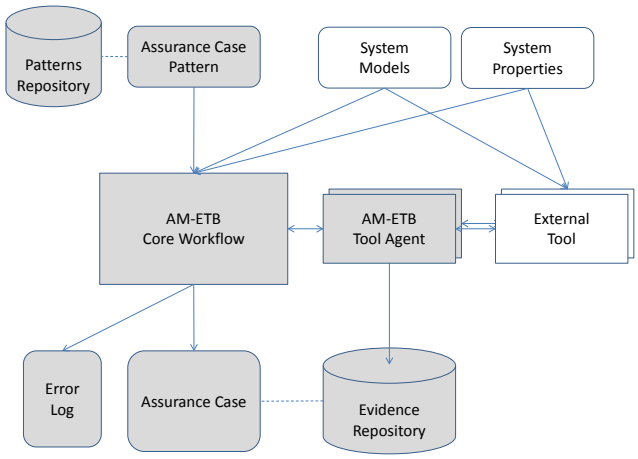

Figure 5: AM-ETB Architecture

to actual system models and properties. The result is a regular assurance case for the system model under consideration which gather all the valid evidence. AM-ETB provides a concrete user-friendly textual representation for the patterns as well as a machineable (parsed) representation to be used internally.

Workflow Management. The workflow of the AM-ETB is defined by the instantiation of an assurance case pattern. AM-ETB develops the instantiated goals according to the "supported by" relation within the assurance case pattern. That is, goals and supporting strategies are progressively developed along the argument. Whenever a leaf evidence (node) is reached and instantiated, AM-ETB initiates a specific evidence checking activity and delegates it to the appropriate tool agent. Upon successful completion, AM-ETB produces a fully instantiated assurance case with all evidence validated and recorded for the actual system configuration and/or reconfiguration. If instantiation failed at some point and/or evidence cannot be re-constructed or validated, AM-ETB produces an error report.

More specifically, AM-ETB provides primitives for launching the (instantiation) workflow for an assurance case pattern on a system model and, on termination, provide a reference for the instantiated assurance case and information about its completeness as well as the backlog of all encountered issues related to evidence re-generation.

Assurance Cases. As explained earlier, an assurance case is obtained as the instantiation of the assurance case pattern in the context of a given system model and associated properties. Upon successful construction, the assurance case contains fully instantiated goals and their associated supporting evidence, re-generated and checked. AM-ETB provides APIs allowing the traversal/visiting of the underlying assurance case structure as well specific generators for user-friendly presentations like Goal Structuring Notation (GSN), structured hypertext, etc.

Tool Agents. A tool agent is responsible for constructing and validating a specific category of evidence using specific external

tools/methods/humans. A taxonomy of evidence considered by AM-ETB is presented in [4]. Usually, evidence is either constructed by running a tool/method with some parameters and/or by interacting with a human agent/expert. Once constructed, the evidence needs to be validated, that is, checked if it is appropriately used in the actual context. Again, checking can be done automatically using a tool and/or through interaction with a human agent. AM-ETB and tool agents could interact either synchronously or asynchronously. That is, the AM-ETB may either wait until the request is processed by a tool agent or proceed immediately and be notified later about the completion. AM-ETB defines specific tool agent interfaces for starting the agent with actual arguments (system model, etc) and obtaining the results including the *verdict* and the reference to an *explanation* (tool output, extra-document, etc) recorded in the evidence database.

Evidence Repository. The evidence repository is used to record the information produced by tool agents and to make it available to the AM-ETB for inclusion in the final assurance cases. AM-ETB provide all the necessary bookkeeping for interacting with tool agents, namely well-identified placeholders for storing evidence information (tool output, tool trace, document, etc) along with meta attributes (generation time, generation tool, ...) as well as the primitives to retrieve evidence information from specific placeholders.

3.2 AM-ETB Running Example

As an illustrative example, consider the assurance case pattern(s) presented in Figure 6. This pattern formalizes a simple argument for establishing when a MILS architecture policy P is considered *safe*, that is, in the example when it has a deadlock-free behavior and conforms to a number of standards. In turn, deadlock-freedom of a policy is ensured whenever every underlying MILS component (subject S) is deadlock-free as well as their overall composition is also deadlock-free.

This pattern can be instantiated for any MILS policy architecture. The result in Figure 7 shows the instantiation result obtained on policy architecture named A containing two interacting subjects $S1$ and $S2$.

Notice that, along the instantiation of the pattern on the policy architecture, AM-ETB calls the specific tool agents for checking/constructing of the required evidence. In this particular case, three different tool agents are invoked. The first agent is used to check the availability of the ISO certificates required for the policy architecture A and will be invoked twice, for both certificates (iso-xxx, iso-yyy). The second agent is used for checking deadlock-freedom forevery individual subject of the policy, and will be also invoked twice, namely for subjects $S1$ and $S2$. The third agent is used to check the conditions for deadlock-free composition of $S1$ and $S2$, and is invoked once.

4 TOWARDS CERTIFICATION OF ADAPTIVE MILS

It is envisioned that the certification of the Adaptive MILS is standards-based. Accordingly, an Adaptive MILS system may be certified against one or several regulatory standards, depending on the application domain of the system (i.e. "DO-178C for airborne systems), that are incorporated in the requirements of the assurance case.

The evaluation and certification of adaptive systems is based on the analysis of the assurance case and the respective evidence. The system may be certified if the claims in the assurance case are deemed appropriate, the evidence is valid, and the argument is correct. Similarly, the validation and certification of Adaptive MILS systems applies an approach that considers the soundness of the assurance case. We discuss hereafter the aspects considered when determining the soundness of the Adaptive MILS assurance case.

Completeness of the Assurance Case

A first aspect of great importance is the degree to which the assurance case has been completed (fully instantiated). This may be achieved by looking at uninstantiated and undeveloped claims. Undeveloped claims are claims that do not end with evidence nodes to support that claim. The AM-ETB indicates the claims that are undeveloped or uninstantiated subsequent to automatic instantiation of the arguments in the assurance case. In case the assurance case is incomplete, the further development of these claims becomes liabilities for developers to ensure completeness of the assurance case prior to the evaluation of the adaptive MILS system.

The completeness of the assurance case can be reported quantitatively as follows. If the argument structure Arg_j has a total of C claims and C_u is the number of undeveloped claims, then C_u/C is the ratio of undeveloped to total claims. This can be extended to the C different node types of the argument, in isolation or in combination. A finished argument is therefore, one for which the total number of nodes N is the same as the total number of developed and instantiated nodes, $N_{d,i}$ [9].

Sufficiency of the Arguments

The second aspect that needs to be considered when evaluating the soundness of an assurance case is the sufficiency of the argument. The sufficiency of the argument concerns the premises of the argument being strong enough to support the conclusions being drawn, taking into account the requirements that need to be met to satisfy the higher-level claim [15].

The Tool Agents of the AM-ETB interact with external tools that constitute methods such as Model-Checker, Satisfiability Modulo Theories (SMT) solvers and theorem provers to analyse the overall argument of a safety case. In addition, developing the assurance case based on regulatory standards may increase the sufficiency of the argument [17].

Sufficiency of the Evidence

The sufficiency of evidence refers to the extent to which the evidence supports the arguments, the integrity of the evidence and fulfilment of evidence requirements. As the collection and maintenance of the evidence is the responsibility of the AM-ETB, it indicates whether all evidence is present and notifies evaluators when evidence is updated. The integrity of evidence additionally contributes to the sufficiency of the evidence (i.e. "in the case of tool-derived evidence, tool qualification and assurance become important issues" [15]).

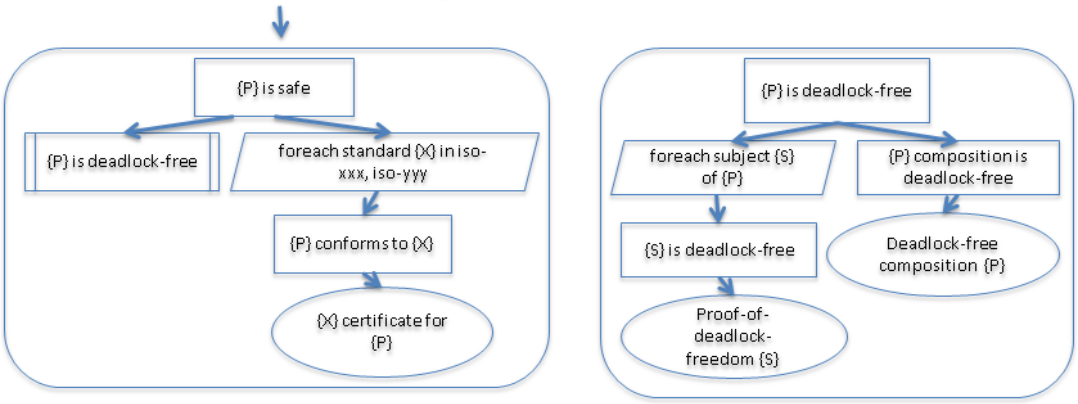

Figure 6: An assurance case pattern in AM-ETB

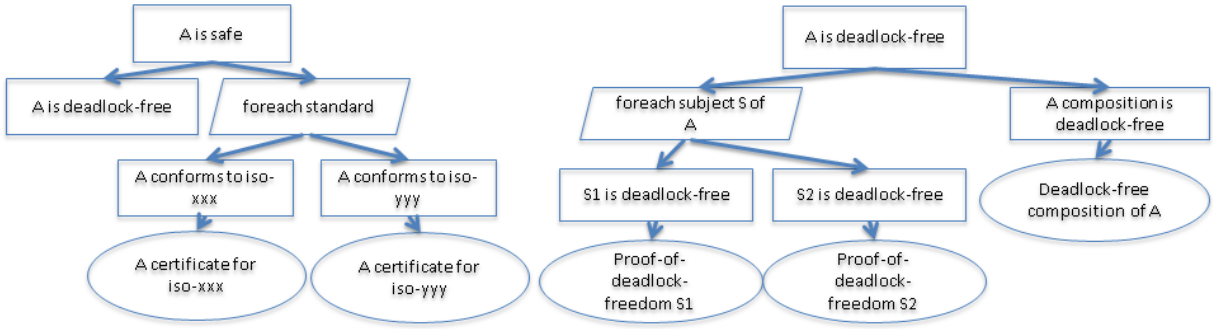

Figure 7: The assurance case obtained by automatic instantiation

Sufficiency of the Assumptions

The AM-ETB incorporates theorem provers and model checkers that refuse to sign off on a proof until all necessary assumptions have been formulated appropriately and included in the statement of the theorem. Subsequently, the assumptions are to be manually analysed and evaluated by evaluators or certifiers by "considering whether they can be substantiated by subsidiary arguments and evidence, in what circumstances they might be invalidated (based on fault tree analysis), what the consequences are if the assumption is false (failure modes) and by monitoring the assumptions during runtime" [16].

5 CONCLUSION

In this paper, we presented a vision on how to certify Adaptive MILS systems. According to this vision, developed in the context of the CITADEL project, the safety and security of an Adaptive MILS systems is evaluated and certified by employing assurance cases. The latter provides arguments, supported by a body of evidence, to justify a set of claims concerning the properties of the system.

A set of assurance case argument patterns was designed to be used for the construction of Adaptive MILS assurance cases. The argument patterns define requirements for the necessary information to instantiate the arguments, and evidence to support the claims made as part of the arguments. The instantiation of assurance case arguments relies on an implementation of the concept of an evidential tool bus, the AM-ETB and system models in the Adaptive MILS Modelling and Specification Language [3].

The AM-ETB gathers the evidence from leaf nodes into a database, and builds the assurance case by establishing higher level goals and the chain of reasoning expressed by assurance case patterns. The output of the AM-ETB comprises the argument model, which together with the evidence builds the assurance case.

The assurance case architecture proposed in this work is modular in its nature, which means that assurance cases may be built and evaluated both top-down as well as a bottom-up. A top-down approach divides each claim into components whose conjunction implies the claim, and recurse down to subclaims supported by evidence. Conversely, a bottom-up approach regards each subclaim, supported by evidence, as an independent unit and conjoins these to produce higher-level subclaims that combine recursively to deliver the top claim.

The assurance case is evaluated, and the system may be certified if the claims are deemed appropriate, the evidence deemed valid and the argument correct. The soundness of the assurance case is determined by considering the completeness of the assurance case, the sufficiency of the argument, the sufficiency of the evidence, and the sufficiency of the assumptions. The security, safety, and other key properties of the Adaptive MILS system are verified on the basis of the soundness of the argument and validity of the evidence supporting each of the arguments in the assurance case.

REFERENCES

[1] David Benyon. 1993. Adaptive systems: A solution to usability problems. *User Modeling and User-Adapted Interaction* 3, 1 (01 March 1993), 65–87. https://doi.org/10.1007/BF01099425

[2] S. Bhattacharyya, D. Cofer, D. Musliner, J. Mueller, and E. Engstrom. 2015. Certification Considerations for Adaptive Systems. In *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS)*.

[3] CITADEL-D3.1 2017. *CITADEL Modeling and Specification Languages*. Technical Report D3.1, Version 1.0. CITADEL Project.

[4] CITADEL-D5.1 2017. *Interfaces and Workflow Definition for AM-ETB*. Technical Report D5.1, Version 1.0. CITADEL Project.

[5] CITADEL-WWW [n. d.]. The CITADEL Project Web Site. ([n. d.]). http://www.citadel-project.org/.

[6] Dan Colesniuc. 2013. Cyberspace and Critical Information Infrastructures. *Informatica Economica* 17, 4 (2013), 123 – 132.

[7] EURO-MILS Consortium. 2015. EURO-MILS Common Criteria Protection Profile Whitepaper. (2015). http://www.euromils.eu/downloads/Deliverables/Y2/2015-EURO-MILS-Protection-Profile-White-Paper-V1.2.pdf.

[8] Simon Cruanes, Gregoire Hamon, Sam Owre, and Natarajan Shankar. 2013. Tool Integration with the Evidential Tool Bus. In *VMCAI*.

[9] D-MILS-D2.4 2014. *Assurance arguments for AADL error models and MILS-AADL formal translations*. Technical Report D2.4, Version 1.0. D-MILS Project. Available at http://www.d-mils.org/page/results.

[10] D-MILS-D4.2 2014. *Compositional assurance cases and arguments for distributed MILS*. Technical Report D4.2, Version 1.0. D-MILS Project. http://www.d-mils.org/page/results.

[11] D-MILS-WWW [n. d.]. The D-MILS Project Web Site. ([n. d.]). http://www.d-mils.org/.

[12] GSNstandard 2011. *GSN Community Standard*. Technical Report. Origin Consulting (York) Limited. http://www.goalstructuringnotation.info/documents/GSN_Standard.pdf

[13] O.C.L. Haas and K.J. Burnham. 2008. *Intelligent and Adaptive Systems in Medicine*. Taylor & Francis.

[14] George A. Ditzel III and Paul Didier. 2015. Time Sensitive Network (TSN) Protocols and use in EtherNet/IP Systems. In *2015 ODVA Industry Conference and 17th Annual Meeting*. Frisco, Texas, USA.

[15] Tim Kelly. 2007. Reviewing Assurance Arguments - A Step-By-Step Approach. In *DNS Workshop 2007*. T.P. Kelly. Available at https://www-users.cs.york.ac.uk/tpk/dsnworkshop07.pdf.

[16] John Rushby. 2009. A Safety-Case Approach for Certifying Adaptive Systems. In *In AIAA Infotech Aerospace Conference, American Institute of Aeronautics and Astronautics, John Rushby*. Available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.159.905&rep=rep1&type=pdf.

[17] John Rushby. 2010. Formal Methods and Argument-Based Safety Cases. (2010). Available at http://www.csl.sri.com/users/rushby/slides/marktoberdorf10.pdf.

[18] John Rushby. 2015. *The Interpretation and Evaluation of Assurance Cases*. Technical Report. Computer Science Laboratory, SRI International, Menlo Park, CA. Available at http://www.csl.sri.com/users/rushby/papers/sri-csl-15-1-assurance-cases.pdf.

[19] John M. Rushby. 2005. An Evidential Tool Bus. In *Formal Methods and Software Engineering, 7th International Conference on Formal Engineering Methods, ICFEM 2005, Manchester, UK, November 1-4, 2005, Proceedings*. 36–36. https://doi.org/10.1007/11576280_3

[20] Douglas Warfield. 2012. Critical Infrastructures: IT Security and Threats from Private Sector Ownership. *Information Security Journal: A Global Perspective* 21, 3 (2012), 127–136. https://doi.org/10.1080/19393555.2011.652289

[21] S. Yamamoto and Y. Matsuno. 2013. An evaluation of argument patterns to reduce pitfalls of applying assurance case. In *2013 1st International Workshop on Assurance Cases for Software-Intensive Systems (ASSURE)*. 12–17. https://doi.org/10.1109/ASSURE.2013.6614265