Original Article
https://doi.org10.56094/jss.v58i1.215

International System Safety Society

www.jsystemsafety.com

Journal of System Safety

Established 1965 Vol. 58 No. 1 (2023)

Incremental Assurance Through Eliminative Argumentation

Simon Diemert[ab], John B. Goodenough[c], Jeff Joyce[d], Charles B. Weinstock[c]

[a] *Corresponding author email: simon.diemert@cslabs.com*
[b] *Critical Systems Labs, University of Victoria; Victoria, Canada*
[c] *Carnegie Mellon Software Engineering Institute; Pittsburgh, United States*
[d] *Critical Systems Labs; Vancouver, Canada*

Keywords
assurance cases, confidence, eliminative argumentation, goal structuring notation

Peer-Reviewed
Gold Open Access
Zero APC Fees

Online: 22-Feb-2023

Cite As:
Diemert S. et al, Incremental Assurance Through Eliminative Argumentation. Journal of System Safety. 2023;58(1):7-15. https://doi.org10.56094/jss.v58i1.215

ABSTRACT

An assurance case for a critical system is valid for that system at a particular point in time, such as when the system is delivered to a certification authority for review. The argument is structured around evidence that exists at that point in time. However, modern assurance cases are rarely one-off exercises. More information might become available (e.g., field data) that could strengthen (or weaken) the validity of the case. This paper proposes the notion of incremental assurance wherein the assurance case structure includes both the currently available evidence and a plan for incrementally increasing confidence in the system as additional or higher quality evidence becomes available. Such evidence is needed to further reduce doubts engineers or reviewers might have. This paper formalizes the idea of incremental assurance through an argumentation pattern. The concept of incremental assurance is demonstrated by applying the pattern to part of a safety assurance case for an air traffic control system.

INTRODUCTION

Assurance cases are important engineering artifacts used to demonstrate that a critical system is acceptably safe or secure; they are comprised of a structured argument supported by evidence generated throughout the system's development lifecycle (Kelly, 1998; Assurance Case Working Group, 2021). Assurance cases are required for compliance with standards such as ISO 26262, UL 4600, and EN 50126, and are necessary for regulatory submissions in some jurisdictions. Assurance cases adopt a goal-oriented approach to assurance that is suitable for development of modern systems where the reasons why the system is safe or secure are complex.

Evidence is an essential ingredient of an assurance case. Without evidence, the arguments therein cannot be substantiated. However, while developing an assurance case a question that often arises is: what evidence is needed to support the case? Since assurance cases have a role to play across all phases of systems development, there are many contexts where this question is relevant and in each context the answer might be different. For example, during early prototyping of a system the evidence necessary to convince stakeholders that the system will eventually achieve its safety or security objectives is different from the evidence presented to an assessor during regulatory review. Regardless, in all contexts the evidences' role is to support an argument that aims to convince a reader (management, business partners, assessors, or the public) that the system has achieved the identified safety or security objectives.

Viewing an assurance case's purpose as one of persuasion allows the question posed above to be re-framed into one of confidence: what evidence is needed to be confident that a claim in the assurance case is valid? From this perspective, not all evidence is the same and different pieces of evidence will contribute by varying degrees to confidence in a claim. For example, for a software routine, a formal proof is typically considered to be more convincing than test results for establishing that the routine is defect free.

To complicate matters, not all evidence is available when an assurance case is prepared. Assurance cases are valid at a specific point in time, such as the day the system is delivered to a certification authority for review. As a result, the argument is structured around what evidence is expected at that point in time, as if this is the final state of what will ever be known about the system by engineers responsible for assuring the system. However, modern assurance cases are rarely one-off exercises and stakeholders may anticipate that more information will become available, such as field data, that could strengthen (or weaken) the validity of the assurance case argument (Koopman & Wagner, 2020). This means that confidence in the assurance case will change (and hopefully increase) as further evidence becomes available.

Since different types of evidence, each carrying a different weight in terms of confidence they afford, become available at different times it follows that confidence in the assurance case changes incrementally over time. As each piece of evidence becomes available, confidence in the top-level claim of the case increases (or decreases) by some incremental amount. This paper applies this notion of incremental assurance to the existing Eliminative Argumentation (EA) method for developing confidence in an assurance case (Goodenough, Weinstock, & Klein, 2015). The idea of incremental assurance is formalized as an argumentation pattern that may be employed in any EA-style assurance case. The pattern is applied to an assurance case fragment for an air traffic control system to illustrate incremental assurance in a real-world use of EA.

ASSURANCE CASES AND ELIMINATIVE ARGUMENTATION

An assurance case is a structured argument showing why one should have confidence in the validity of a claim given certain evidence. By providing explicit claims and reasoning for why evidence is believed to support the claims, an assurance case makes explicit the reasoning that is otherwise often implicit in arguments intended to show that a system is acceptably safe or secure (or any other property of interest). The assurance case has its roots in the notion of a safety case, which is used in safety critical system development. This section provides a brief introduction to the Eliminative Argumentation (EA) method for developing assurance cases.

THE ROLE OF DOUBT IN SAFETY ARGUMENTATION

Safety arguments that aim to directly "prove" that a system is safe are subject to confirmation bias. The fatal crash of the Nimrod military aircraft in Afghanistan in 2006 is a well-known of example of how confirmation bias can undermine an assurance case. The post-crash investigation found that: "the Nimrod Safety Case [was] fatally undermined by an assumption by all the organisations and individuals involved that the Nimrod was 'safe anyway', because the Nimrod fleet had successfully flown for 30 years, and they were merely documenting something which they already knew. … The Nimrod Safety Case became essentially a paperwork and 'tickbox' exercise" (Haddon-Cave, 2009).

An assurance case starts with a top-level claim which is recursively decomposed into sub-claims which are eventually supported by evidence. Traditional notations (e.g., GSN) do not emphasize the expression of doubt and therefore it is less likely that the claims and evidence expressed will be questioned. There is no way to express residual risk.

In practice, engineers have many reasons to doubt the safety of a system. Doubting oneself and subsequently addressing those doubts with further claims and evidence is central to the scientific and engineering approach to problem solving. But enumerating doubts is not, on its own, sufficient to mitigate confirmation bias. Enumeration of doubt only shifts the question from "Do the sub-claims completely support the top-level claim?" to "Have all doubts been identified?" However, this question, in turn, necessitates further argument explaining why one believes that there is, at most, a slight possibility that a relevant doubt has been overlooked. For example, one might doubt the completeness of a set of failure modes derived for a component in the system based on a Failure Modes and Effects Analysis (FMEA). An argument countering this doubt might claim that a combination of experienced persons and systematic methodology provide confidence in the completeness of the failure modes. Even so, a residual doubt will exist and be communicated to stakeholders as a risk associated with the component.

PRIMER ON ELIMINATIVE ARGUMENTATION

The question arises: Why should we believe an assurance case? A lack of confidence in a claim implies that there are doubts that it is true. If doubts exist, we cannot be completely confident in the claim. Every time a doubt is resolved (i.e., minimized or eliminated), confidence in the claim increases. When all doubts have been sufficiently resolved, there is high confidence in the claim. The practice of postulating and resolving doubts about an assurance argument is the basis for Eliminative Argumentation (EA) introduced by Goodenough et al. as an adaption of Toulmin's and Kelly's notation (Goodenough, Weinstock, & Klein, 2015; Kelly, 1998; Toulmin, 2003). EA provides a framework for constructing an argument and assessing confidence in the argument based on the identification and eventual resolution of doubts[1].

EA addresses confirmation bias by including the notion of doubt as a first-class citizen. In EA, these doubts are called "defeaters" in the sense that they defeat aspects of an argument. There can be doubts that rebut claims (e.g., "the valve will open" might be rebutted by "unless the valve is stuck"), undermine evidence (e.g., "all tests pass" might be undermined by "but the system tested is not the one deployed"), or undercut inferences (e.g., "if all the doubts about the claim have been resolved, then the claim is true" might be undercut by the defeater "unless there is an unidentified doubt that should have been considered"). When defeaters describe doubt that is considered acceptable as residual doubt (without further argument), then they are marked as "residual" and contribute to the overall residual doubt associated with the case.

For an example of the EA notation, consider a case arguing that it will snow in Vancouver, Canada tomorrow shown in Figure 1. A strategy node (S0002) describes the argumentation strategy. The strategy has two child defeaters that challenge the top-level claim. D0003 suggests that there might be insufficient moisture to cause snow. The presence of a storm over the Pacific Ocean near Vancouver resolves this defeater (C0006). Two defeaters rebut this claim: it is possible that the wind will change direction (D0007), or the storm will not reach Vancouver (D0008). Further evidence (E0009 and E0012) is presented to resolve these defeaters. But each piece of evidence is undermined by additional doubt (D0010 and D0013). Neither of these undermining defeaters are resolved and therefore represent residual doubt in the case (annotated as "Res"). D0004 suggests that the temperature might not drop low enough to cause snow; however, this doubt is resolved by evidence (E0015) which is accepted without further doubt (annotated as "OK"). An inference rule (IR0005) relates the two defeaters (D0003 and D0004) to C0001. Defeater D0017 undercuts the rule by suggesting that it is unsound. Evidence E0018 is presented to resolve the undercutting defeater. The

[1] *In many cases it is not possible to completely eliminate doubt. This paper uses "resolve" to indicate that doubt is reduced to a low enough level that it is accepted as not significant enough to challenge the validity of the case.*

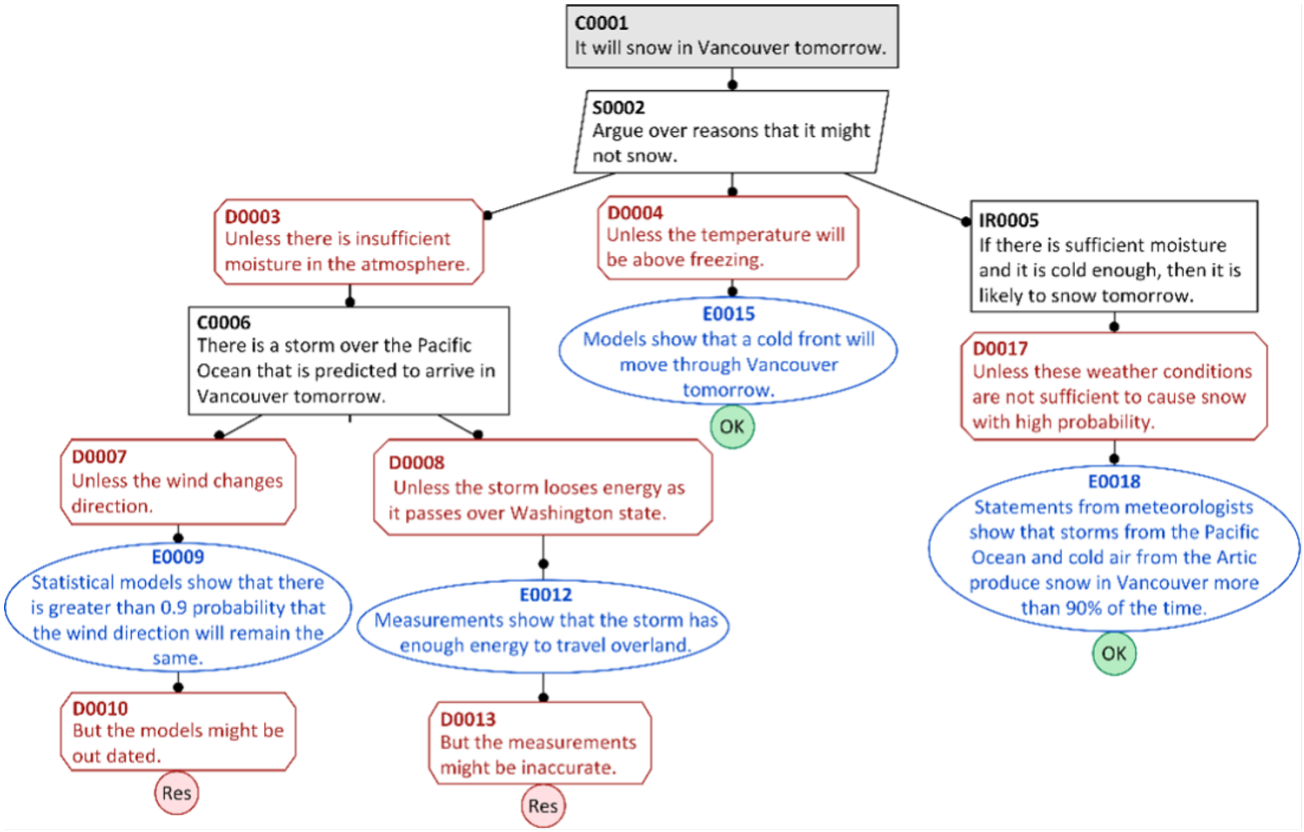

Figure 1: Sample EA case for arguing that it will snow in Vancouver.

EA notation also includes context, assumption, and undeveloped nodes, not shown in Figure 1[2].

INCREMENTAL ASSURANCE

In an EA assurance case, confidence in the top-level claim is established by showing that reasons to doubt the case have been resolved. Of course, it is difficult to eliminate doubt with absolute certainty. Instead, authors of assurance cases present enough evidence to give the reader sufficient confidence the doubt is resolved. This necessitates that the author (and in turn the reader) of the assurance case make a judgement about the level of confidence provided by each piece of evidence and the inference rule. Each piece of evidence added to the assurance case incrementally increases confidence that a doubt has been resolved and thus increases overall assurance in the system.

This notion of incremental assurance is implicitly used in functional safety standards such as IEC 61508, ISO 26262, DO-178C, and EN 50126. These standards employ a level-of-rigor approach whereby confidence in the safety integrity of a system is increased by prescribing more demanding engineering activities. For example, per ISO 26262 Part 6, for software assigned Automotive Safety Integrity Level (ASIL) A (lowest criticality) fault injection testing is recommended; however, for software assigned ASIL D (highest criticality) fault injection testing is a highly recommended activity. In other words, the level of assurance is incrementally increased as additional doubts are identified and resolved with appropriate evidence. Functional safety standards also prescribe the minimum criteria for accepting evidence as sufficient for the purpose of resolving implicit doubts about the safety of a system.

[2] *This example and others in this paper were prepared using Socrates – Assurance Case Editor, a web-based tool for collaborative assurance case development and maintenance that supports the EA and GSN notations. See https://safetycasepro.com for more information on the Socrates tool.*

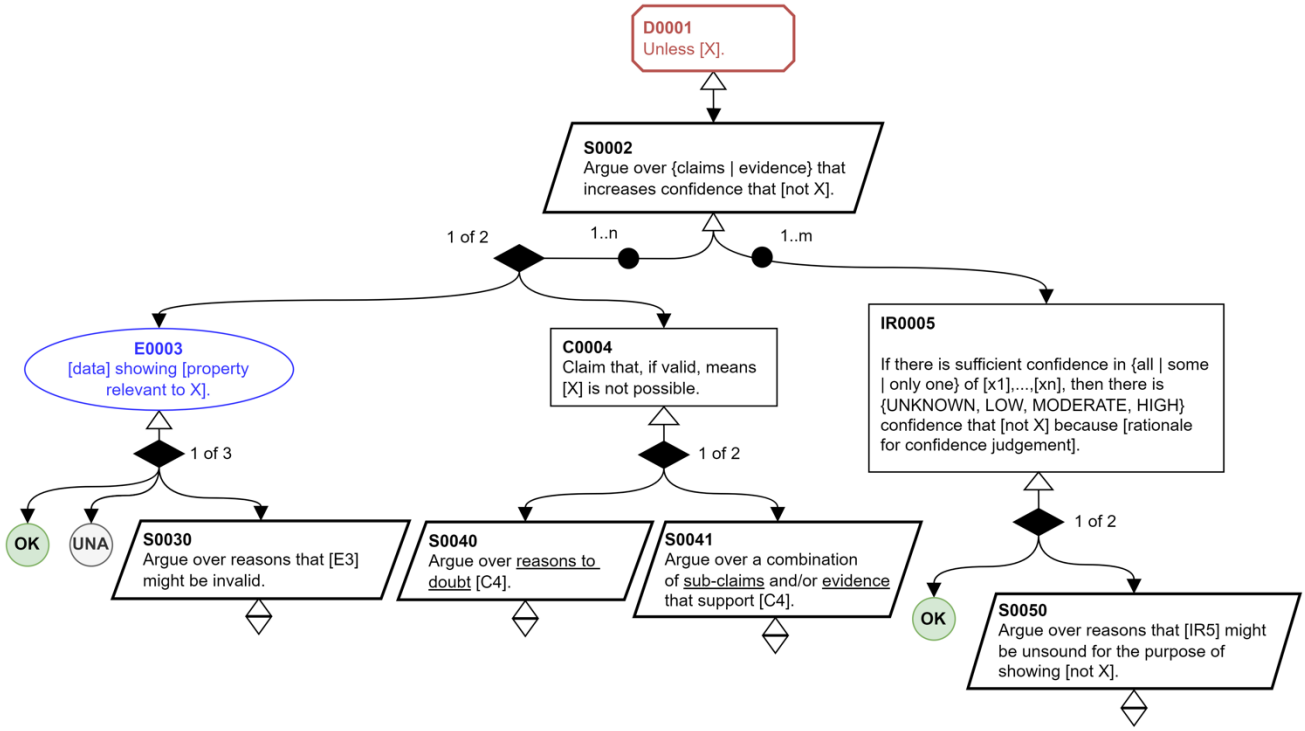

Figure 2: Specification of pattern for incremental assurance using Eliminative Argumentation.

This section presents the main contribution of this paper: an EA reasoning pattern for incremental assurance. While many assurance case patterns exist for GSN-style assurance cases (Kelly, 1998; Szczygielska & Jarzebowicz, 2017), to the authors knowledge, this is the first published pattern using EA. The pattern is a type of a "syntactic pattern" that describes how to correctly express the idea of incremental assurance using the EA notation. This kind of syntactic pattern differs in character from argumentation patterns that describe the system's function. The pattern explicitly describes how individual pieces of evidence, or claims that are ultimately supported by evidence, may be combined to achieve confidence that a defeater in an EA argument is resolved. An essential idea in the pattern is to explicitly describe a confidence calculation in an inference rule that gives the degree of confidence that the defeater is resolved.

The pattern for incremental assurance is depicted in Figure 2 using the pattern specification notation described in the 3rd Edition of the GSN Community Standard (Assurance Case Working Group, 2021). Additionally, within the pattern specification Figure 2 uses braces "{…}" to denote choices for the author and square braces "[…]" denote wording or values that should be populated when the pattern is instantiated. The wording in the pattern specification is generic: some adjustments to wording are required upon instantiation.

The pattern is rooted in a defeater (D0001) that describes a doubt [X] about an arbitrary parent node in the argument. The pattern is applicable regardless of whether the root defeater (D0001) is rebutting a parent claim, undermining evidence, or undercutting an inference rule. Following the style of EA, either claims or evidence (or both) may be presented against the root defeater. The rationale for combining the claims and evidence to address the root defeater is captured in one or more inference rules (IR0005). Critically, each inference rule describes the level of confidence that the root defeater is resolved when there is sufficient confidence in a combination of the claims (C0003) and evidence (E0004). Multiple inference rules should be used to express varying degrees of confidence that might arise from different combinations of claims/evidence; for example, see inference rules IR0026 and IR0034 in Figure 4.

The pattern uses a qualitative assessment of confidence with an ordinal scale: UNKNOWN confidence, LOW confidence, MODERATE confidence, or HIGH confidence. A qualitative

assessment is used (as opposed to a quantitative measure) since assessment of confidence arising from a combination of claims and evidence is typically the subject of engineering judgement based on project context, experience, and best practices described in technical standards, such as IEC 61508 and ISO 26262. When instantiating the pattern, the author(s) of the assurance case should provide a rationale for why a particular combination of claims and evidence gives the indicated level of confidence. The inference rule, though expressed in natural language, is in fact a confidence calculation that indicates how to propagate confidence through the assurance case's argument structure. The pattern depicted in Figure 2 gives a template for a simple instance of such a confidence calculation that requires a minimum number of evidence (or claims) be sufficiently substantiated. However, more sophisticated calculations are possible. In practice, if the calculation is too complex to fit within the confines of box in a diagram, it can be expressed in narrative text accompanying the assurance case.

The pattern specification in Figure 2 provides an opportunity to include further argumentation under the evidence (E0003), claim (C0004), or inference rule (IR0005). This is shown using strategy nodes (S0030, S0040, S0041, S0050) which provide "hooks" to continue to develop the argument structure using EA. Under the evidence (E0003), reasons to doubt the evidence may be captured as undermining defeaters (S0030). Under the claim (C0004), reasons to doubt the claim may be captured as rebutting defeaters (S0040) or supporting claims and evidence may be presented (S0041). Note that in Goodenough et al.'s original formulation of EA, claims could not be supported by sub-claims; however, in practice it is useful to be able to decompose a complex claim into sub-claims that can be more easily argued. Finally, there might be reasons to doubt the soundness of the inference rule used to combine the claims and evidence; these are expressed as undercutting defeaters (S0050).

In addition to evidence or claims advanced to resolve the defeater, it is possible that counter-evidence is available that strengthens the credibility of the defeater. For instance, reports from field trials of a software system might describe a particular misbehavior of the software that increases the credibility of the defeater. For a general discussion of counter-evidence in EA arguments see Goodenough et al.'s report on EA (Goodenough, Weinstock, & Klein, 2015). In the context of this pattern, counter-evidence may be included by presenting it as an instance of E0003 in the pattern and using an inference rule (IR0005) to indicate how the existence of the counter-evidence modifies the confidence that the root defeater (D0001) has been addressed.

In Figure 2, the inference rule (IR0005) indicates how to combine claims and evidence to address the root defeater. However, it depends on "sufficient confidence" (or some other criteria chosen by the author as part of their confidence calculation) being established in the validity of the rule's premises. The pattern requires the user to determine the level of confidence afforded to combinations of claims and evidence. In this regard, there are several cases to consider. First, the case where the evidence (E0003) is not available (marked as "UNA") or where the claim (C0004) is undeveloped is clearly insufficient. Second, the case where evidence (E0003) and/or a claim (C0004) has unaddressed defeaters indicating residual doubt associated with them is more difficult: how much residual doubt can be tolerated without changing the level of confidence indicated by the

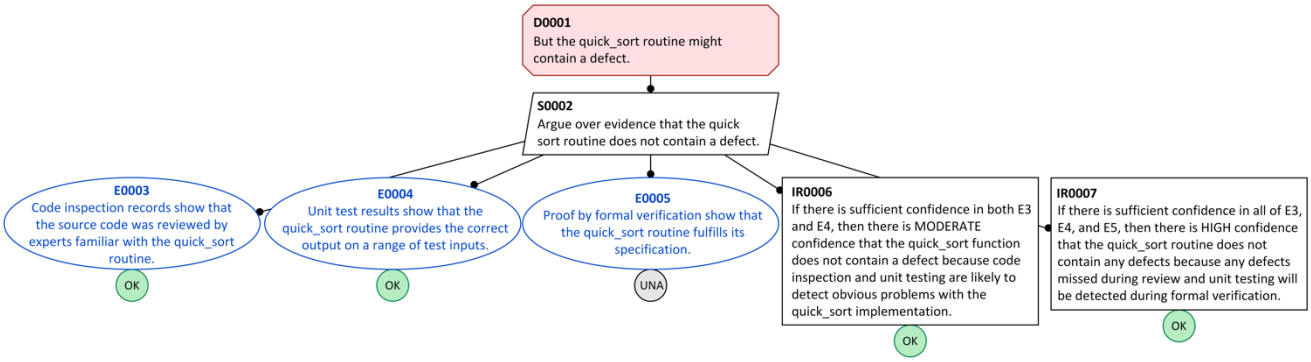

Figure 3: Example of pattern instantiation for a quick_sort routine.

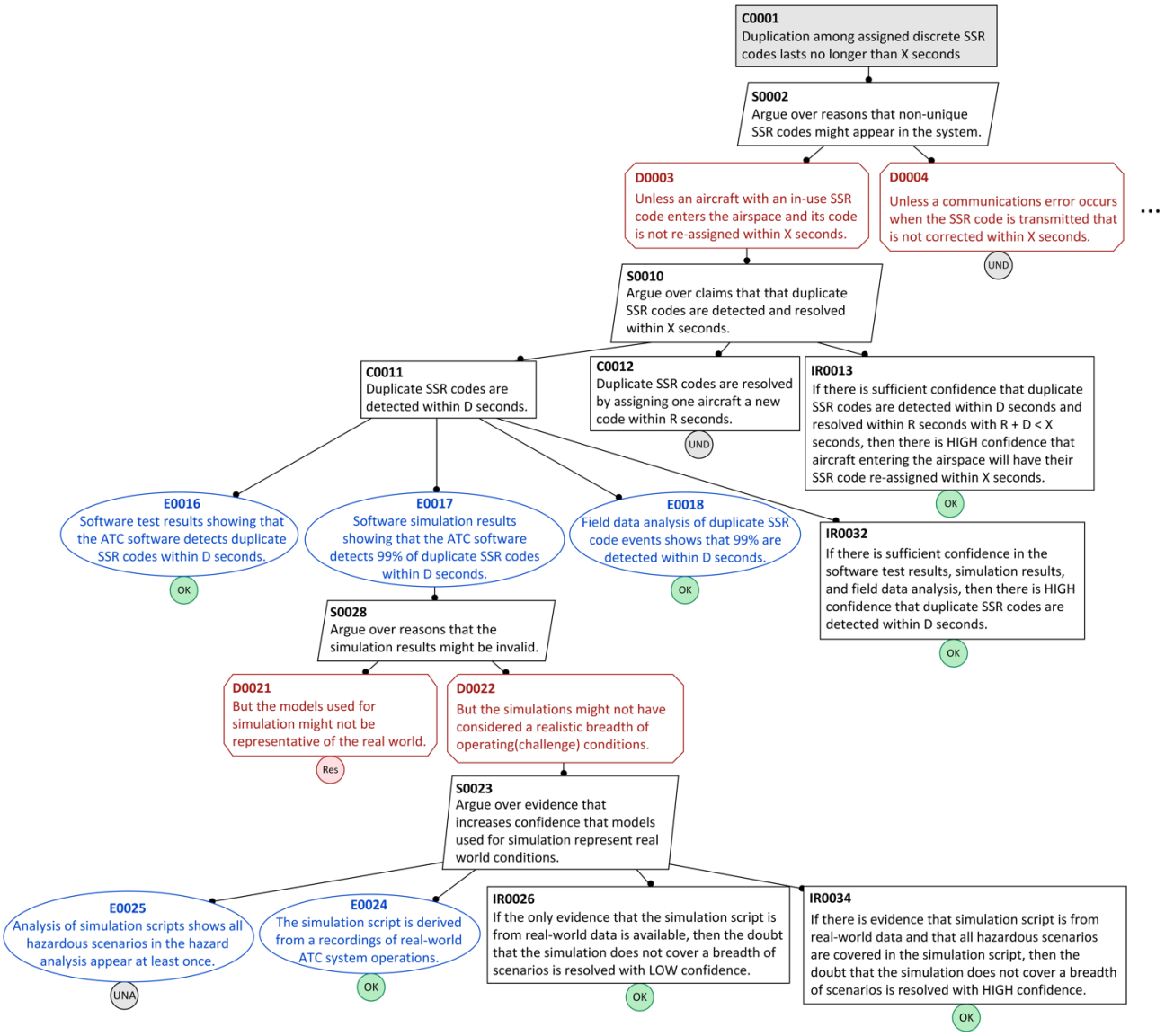

Figure 4: Example of applying the pattern to an air traffic control system.

inference rule (IR0005)? One might (somewhat naively) suggest that zero residual doubt is tolerable in the inference rule. However, in real-world systems there is almost always residual doubt that cannot be resolved, and it is therefore not practical for the pattern to demand zero residual doubt. The confidence calculation in the inference rule (IR0005) should incorporate the tolerable level of residual doubt in the claims and evidence advanced against the root defeater. Formalizing the calculation to combine varying degrees of confidence in the supporting claims and evidence is a topic of on-going research.

To illustrate the core idea(s) of incremental assurance, the pattern is instantiated as an argument fragment related to a quick_sort software routine, see in Figure 3. The root of the fragment is a defeater (D0001) describing a doubt that the software routine has a defect. Three pieces of evidence are proposed to address this defeater: code inspection records (E0003), unit test results (E0004), and a proof using formal verification techniques (E0005). The code inspection and unit test results are marked as "OK" indicating they are available and determined to be acceptable. However, the formal proof is marked as unavailable ("UNA"). Two inference rules (IR0006, IR0007) describe how to logically combine the evidence to address the root defeater, these are instances of IR0005 from Figure 2. These rules reflect

widely accepted software quality assurance practices. If there is sufficient confidence in the code inspection and unit testing then there is moderate confidence that the routine is defect free; however, there cannot be high confidence because testing and inspection cannot absolutely prove the absence of software defects (IR0006). If the formal proof is added, then confidence is increased (IR0007). In this example, since only the inspection and unit test results are available, the authors of the assurance case may conclude that the root defeater is addressed with moderate confidence. When a formal proof becomes available, then the level of confidence can be incrementally increased.

APPLICATION TO AN ASSURANCE CASE FOR AN AIR TRAFFIC CONTROL SYSTEM

The pattern for incremental assurance was applied to a fragment of an assurance argument for an air traffic control system. This application is motivated by direct experience with the development of a safety assurance case by Raytheon Canada for an air traffic management system delivered to Canada's air navigation services provider, Nav Canada. The argument fragment addresses duplicate discrete SSR code events that occur when a discrete SSR code, the unique identifier assigned to an aircraft by air traffic control, is used by more than one aircraft in the same airspace. Since the SSR code is used as a "primary key" by air traffic management software, duplication events could contribute to a hazardous loss of minimum separation distance between two aircraft (for example, a duplicate SSR code event could cause radar altitude data for one aircraft to displayed as the current altitude for a different aircraft).

In practice, duplicate SSR code events are not necessarily rare but should normally be resolved by air traffic controllers and pilots before they become serious concerns. The argument fragment in Figure 4 argues that discrete SSR codes assigned to aircraft in the controlled airspace are unique over a tolerable interval of X seconds (C0001). For illustrative purposes, two defeaters are advanced against this claim: aircraft arriving from another airspace might be using a code that is already in-use in the current airspace (D0003), or communication errors between controllers and pilots might contribute to duplication events (D0004); other reasons to doubt the top-level claim exist but are not listed for brevity.

Defeater D0003 and its descendants S0010, C0011, C0012, and IR0013 are the first instance of the pattern described above in Figure 2. In this instance, the possibility that an aircraft enters the airspace with a duplicate SSR code is addressed by a combined strategy of detecting the duplicate code (C0011) and resolving the duplication (C0012) within a duration of time that is less than the tolerable interval X. The inference rule shows that if this "detect and resolve" strategy is shown (by further argumentation) as valid with sufficient confidence, then there is high confidence that the duplicate SSR code events arising from incoming aircraft are resolved within a tolerable duration (IR0013). Further argumentation is provided for Claim C0011.

Claim C0011 is supported by three pieces of evidence: software test results (E0016), simulation results (E0017), and historical field data analysis (E0018). This evidence in combination with the inference rule (IR0032) forms a structure that is similar to the pattern. In particular, the inference rule (IR0032) describes how to combine the evidence (E0016-E0018) to support a parent (C0011). However, instead of establishing confidence that a doubt is resolved, this inference establishes confidence that Claim C0011 is valid. In turn, confidence in Claim C0011 resolves the parent defeater (D0003) via Inference Rule IR0013.

The simulation results described by Evidence E0017 are undermined by two additional defeaters: the models used for simulation might not represent the real-world system (D0021), and the simulation studies might not have covered a sufficient breadth of operational conditions (D0022). For this example, Defeater D0021 is left undeveloped. However, D0022 is addressed using another instance of the pattern for incremental assurance. In this instance, two additional pieces of evidence are listed. First, that Evidence E0024 says that simulation scripts are derived from real-world operations. Second, Evidence E0025 says that the simulation scripts cover the identified hazardous scenarios. Two inference rules are given that indicate how to combine these pieces of evidence together to assess the confidence that Defeater D0022 is resolved. According to the Inference Rule IR0026, if Evidence E0025 is not available, then there is at best "low confidence" that the simulations cover a range of scenarios. However, per Inference Rule IR0034, if a coverage analysis is available, then there is "high confidence" that Defeater D0022 is resolved.

In Figure 4, note that Evidence E0025 is marked as unavailable (UNA). Then only Inference Rule IR0024 applies and there is low confidence that the simulations cover a breadth of operating conditions (D0022). This low confidence in E0025 is propagated through the argument using the Inference Rules IR0032 and IR0013. Overall, even disregarding the undeveloped (UND) Claim C0012 and the residual (RES) Defeater D0021, it is concluded that Defeater D0003 cannot be resolved ("eliminated") with high confidence because there is not sufficient confidence in Claim C0011, i.e., that duplicate SSR codes are detected in a timely way.

CONCLUSION AND FUTURE WORK

Assurance cases are important engineering artifacts produced for safety and security-critical systems. As a system progresses through the lifecycle additional evidence may become available that increases confidence in the assurance case. This paper introduced the concept of "incremental assurance" in which the confidence in an assurance case, expressed using the Eliminative Argumentation method, is increased by providing additional evidence or argumentation that resolves doubts expressed as defeaters within the case. An essential idea is using inference rules to explicitly describe the change in confidence afforded by additional evidence or claims that a doubt is resolved. The concept of incremental assurance was described as an argumentation pattern, which is also the first pattern to employ the Eliminative Argumentation method. The pattern was illustrated by applying it to an argument fragment for an air traffic control system. Future work in this area will identify additional patterns for incremental assurance and extend the idea of confidence propagation using inference rules.

ACKNOWLEDGEMENTS

This material is based upon work funded and supported by Critical Systems Labs Inc. and the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

REFERENCES

[1] Assurance Case Working Group. (2021). Goal Structuring Notation Community Standard - Version 3. Safety-Critical Systems Club.

[2] Goodenough, J. B., Weinstock, C. B., & Klein, A. Z. (2015). Eliminative Argumentation: A Basis for Arguing Confidence in System Properties. Pittsburgh, Pennsylvania: Software Engineering Institute, Carnegie Mellon University.

[3] Haddon-Cave, C. (2009). The Nimrod Review. London, UK: London Stationary Office.

[4] Kelly, T. P. (1998). Arguing safety - A Systematic Approach to Safety Case Management. University of York.

[5] Koopman, P., & Wagner, M. (2020). Positive Trust Balance for Self-driving Car Deployment. Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops. Springer. https://doi.org/10.1007/978-3-030-55583-2_26

[6] Szczygielska, M., Jarzebowicz, A. (2017). Assurance Case Patterns On-line Catalogue. Advances in Dependability Engineering of Complex Systems (pp. 407-417). Springer. https://doi.org/10.1007/978-3-319-59415-6_39

[7] Toulmin, S. E. (2003). The Uses of Argument. Cambridge University Press. https://doi.org/10.1017/CBO9780511840005